

UNIVERSIDADE ESTADUAL
PAULISTA

Faculdade de Ciências – Bauru
Bacharelado em Ciência da Computação

Gabriel Fazzio de Paula

DESENVOLVIMENTO INTERATIVO DE
DESIGN PARA PÁGINAS NA WEB
UTILIZANDO ALGORITMOS GENÉTICOS

UNESP – Bauru
Março de 2015

Gabriel Fazzio de Paula

Desenvolvimento Interativo de Páginas na Web
Utilizando Algoritmos Genéticos

Orientador: Profa. Dra. Simone das Graças Domingues Prado

Monografia apresentada junto à disciplina
Projeto e Implementação de Sistemas II, do
curso de Bacharelado em Ciência da
Computação, Faculdade de Ciências,
Unesp, campus de Bauru, como parte do
Trabalho de Conclusão de Curso.

UNESP – Bauru
Março de 2015

*“A melhor maneira de prever o futuro é
inventá-lo.”*

(Alan Kay)

RESUMO

O objetivo do presente trabalho é abordar algoritmos genéticos interativos como uma ferramenta para o desenvolvimento de um protótipo de design para páginas na *web*, que muitas vezes pode ser considerada uma tarefa complicada e que requer grande inspiração para atingir um resultado satisfatório. Esta vertente da inteligência artificial foi escolhida pois tem a habilidade de gerar resultados muito variados e possibilitar a seleção dos mesmos de maneira a se adequar ao objetivo do usuário. Esta monografia apresenta componentes e métodos que foram utilizados para o desenvolvimento de uma aplicação web, que mostra páginas ao usuário em conjunto com uma ferramenta de avaliação simples e os seleciona conforme a sua preferência, utilizando conceitos dos algoritmos genéticos. O trabalho, também, contém alguns resultados apresentados pela aplicação e o aplicativo desenvolvido cria protótipos adequados, utilizando algoritmos genéticos converge para uma solução adaptada ao gosto do usuário.

ABSTRACT

The objective of this paper is to approach interactive genetic algorithms as a tool for developing a prototype of the design of a web page, which can be considered a tricky task and demanding of great inspiration to achieve a pleasant result. This field of artificial intelligence was chosen because it has the potential to generate diverse results and allows the user to select them, enabling them to adapt to their needs. This monography shows the components and methods that were used on the development of a web application, that shows pages to the user with a built-in tool used to select and modifies the designs so that they mold to the user needs, using the concepts provided by the genetic algorithms. The paper also contains the results presented by the application and the developed application creates suitable prototypes using genetic algorithms that converges to an adapted solution to the users taste.

Lista de Figuras

Figura 1– Diversidade de mariposas	13
Figura 2– Diversidade em joaninhas	14
Figura 3- Seleção natural dos besouros	15
Figura 4- Gregor Mendel.....	16
Figura 5- Fenótipos da ervilha-de-cheiro	17
Figura 6- Tabela de cruzamento entre genes	18
Figura 7- Proporção 9:3:3:1 em labradores	19
Figura 8– Meiose	20
Figura 9- Crossing-over.....	21
Figura 10 - John Henry Holland.....	23
Figura 11- Algoritmo genético	23
Figura 12- Tradução genótipo para fenótipo na biologia e em algoritmos genéticos ...	24
Figura 13 - Roleta viciada	26
Figura 14- Escalonamento sigma	27
Figura 15– Exemplo de Crossover em um AG	28
Figura 16– Mutações	30
Figura 17– Inserção Steady-state.....	31
Figura 18– Processo de requisição/resposta simples	33
Figura 19-Processo de requisição/resposta dinâmico.....	34
Figura 20- Estrutura básica de um código HTML3	36
Figura 21- exemplo de HTML.....	37
Figura 22 - exemplo de uma página com CSS.....	38
Figura 23- O gene da aplicação	44
Figura 24 - O <i>crossover</i> da aplicação	45
Figura 25 - Mutação gerativa.....	46
Figura 26 - Mutação destrutiva.....	47
Figura 27 - Mutação de troca de nós	47
Figura 28 - Mutação de troca de sequência	48
Figura 29- Tela inicial da aplicação	50
Figura 30 - Função criar_html	51
Figura 31 - Função criar_css	52
Figura 32– Tela de avaliação.....	53
Figura 33-Exemplo de marcação.....	54
Figura 34- Código da roleta viciada.....	55
Figura 35- Função de mutação	56
Figura 36- Mutação gerativa.....	57
Figura 37 - Mutação destrutiva.....	58
Figura 38 - Mutação de troca de nós	58
Figura 39- Mutação de troca de sequência	59
Figura 40– Indivíduo inicial	60
Figura 41– Indivíduo final.....	61

Sumário

1	INTRODUÇÃO.....	9
1.1	PROBLEMA.....	10
1.2	JUSTIFICATIVA	10
1.3	OBJETIVOS	10
1.3.1	OBJETIVO GERAL.....	10
1.3.2	OBJETIVOS ESPECÍFICOS	10
1.4	MÉTODO DE PESQUISA.....	11
2	REFERENCIAL TEÓRICO.....	12
2.1	A EVOLUÇÃO.....	12
2.1.1	INTRODUÇÃO.....	Error! Bookmark not defined.
2.1.2	A VARIAÇÃO	13
2.1.3	A SELEÇÃO NATURAL	14
2.2	A GENÉTICA.....	15
2.2.1	INTRODUÇÃO.....	Error! Bookmark not defined.
2.2.2	O EXPERIMENTO DE MENDEL	16
2.2.3	O GENE.....	17
2.2.4	A TEORIA CROMOSSOMICA DA HERANÇA	19
2.2.5	VARIABILIDADE GENÉTICA.....	20
2.2.6	MUTAÇÃO.....	20
2.2.7	CRUZAMENTO GENÉTICO	20
2.3	ALGORITMOS GENÉTICOS	21
2.3.1	COMPUTAÇÃO EVOLUTIVA	21
2.3.2	BREVE HISTORICO DOS ALGORITMOS GENÉTICOS	22
2.3.3	O ALGORITMO GENÉTICO	23
2.3.4	O CROMOSSOMO.....	24
2.3.5	MUTAÇÃO.....	29
2.3.6	INSERÇÃO	30
2.3.7	ALGORITMOS GENÉTICOS INTERATIVOS	31
2.4	A WEB.....	31
2.4.1	INTRODUÇÃO.....	Error! Bookmark not defined.
2.4.2	O HTTP	32
2.4.3	O PROCESSO DE REQUISIÇÃO/RESPOSTA	32
3	MATERIAIS E MÉTODOS.....	36
3.1	HTML	36
3.2	CSS	37
3.3	O PHP	38
3.4	FRAMEWORKS	39
3.5	LARAVEL.....	39
3.6	TWITTER BOOTSTRAP.....	41
4	METODOLOGIA.....	42
4.1	O ALGORITMO DESENVOLVIDO	42
4.2	DEFINIÇÃO DO CROMOSSOMO.....	42
4.3	OS ALELOS	43
4.4	A INICIALIZAÇÃO.....	44
4.5	A SELEÇÃO.....	44
4.6	O CROSSOVER	45
4.7	A MUTAÇÃO	46
4.8	A INSERÇÃO.....	48

4.9	O CRITÉRIO DE PARADA	49
4.10	A APLICAÇÃO DESENVOLVIDA	49
4.10.1	TELA INICIAL	49
4.10.2	A PRIMEIRA GERAÇÃO	50
4.10.3	A FUNÇÃO “CRIAR_HTML()”	52
4.10.4	A FUNÇÃO “CRIAR_CSS()”	52
4.10.5	TELA DE SELEÇÃO.....	53
4.10.6	A MARCAÇÃO	53
4.10.7	ALGORITMO DE SELEÇÃO.....	55
4.10.8	A FUNÇÃO DE MUTAÇÃO	56
1.1.1.	FINALIZAÇÃO	59
1.1.2.	RESULTADOS	59
5	CONCLUSÃO.....	62
	REFERÊNCIAS	63

1 INTRODUÇÃO

A internet na atualidade tem extrema importância nas vidas das pessoas. Usa-se a internet para muitas tarefas, sejam domésticas ou empresariais e elas só tendem a aumentar, com a evolução tecnológica e o crescimento da inclusão digital, a internet se torna imprescindível para o cotidiano. Entretanto apresenta problemas como, por exemplo, o de usabilidade da página, onde o usuário tem dificuldades para localizar em determinado site o que ele procura, muitas vezes gerando prejuízo (VEEN, 2000). A solução para este problema pode ser encontrada em um bom design. Seja uma página que sofra com um design pobre e por tanto não atinge seus objetivos, ou a dificuldade de criar um esquema que atinja o público alvo são consequências graves de um design fraco, e deve-se procurar uma solução (VEEN, 2000). A evolução Darwiniana é um mecanismo de procura e otimização muito robusto e os algoritmos genéticos se baseiam nele. Ao contemplar organismos evoluídos como as células, vírus, e até seres humanos, é possível observar o quão complexo é este processo (Fogel, 1997). Segundo Holland (1992), a ideia dos algoritmos genéticos é usar elementos da evolução para criar resultados ótimos. A opção escolhida para amenizar este problema neste trabalho foi o design evolutivo. Podendo ser utilizado para gerar desde carros, mesas e diversos objetos cotidianos (BENTLEY, 1996) até antenas de alto desempenho para a NASA (LOHN et al, 2004) os algoritmos genéticos veem sendo utilizados para criação de design, pois não são previsíveis e acabam por gerar resultados que muitas vezes humanos não enxergariam e utilizam de um conceito que se provou muito eficiente.

1.1 PROBLEMA

Na atualidade, o design de páginas da web se tornou um elemento crucial para o sucesso delas. Segundo Flanders (2002), criar um design agradável e elegante pode ser uma tarefa muito difícil, já que depende da inspiração do designer, e pode levar tempo para que se consiga obter protótipos que sejam bem aceitos. Existe, também, o problema de clientes terceiros a um designer não ficarem satisfeitos com o protótipo desenvolvido, exigindo com que o projetista tenha que fazer novos modelos, que demandará tempo de inspiração e execução para um novo protótipo.

1.2 JUSTIFICATIVA

Uma aplicação que possibilite a geração, de maneira simplificada, de protótipos para o *web designer* pode amenizar o problema de criar páginas na web com um design agradável, economizando muito tempo e possibilitando ao designer alguns protótipos que seriam difíceis de serem concretizados, além de amenizar o problema de retrabalho, já que gerar protótipos seria simples, podendo contar até com a participação do cliente final.

1.3 OBJETIVOS

1.3.1 OBJETIVO GERAL

O objetivo deste trabalho é o de desenvolver uma aplicação que, explorando os conceitos de algoritmos genéticos, crie protótipos de páginas web que se adaptem ao gosto do usuário a fim de auxiliá-lo a criar um design de uma página de maneira simples e rápida, de uma qualidade estética desejável e inspirar o trabalho de desenvolvedores web.

1.3.2 OBJETIVOS ESPECÍFICOS

Compreender os conceitos de uma aplicação interativa utilizando algoritmos genéticos. A aplicação deve gerar um protótipo que de suporte ao usuário na escolha de um design para uma página na web.

1.4 MÉTODO DE PESQUISA

Para este trabalho foi realizada uma pesquisa bibliográfica a fim de ter um fundamento teórico, e analisar os conteúdos que abrangem web design, usabilidade e algoritmos genéticos de modo que esse conhecimento possa ser aplicado no desenvolvimento da aplicação.

O procedimento da coleta de dados foi realizado primeiramente através de pesquisas bibliográficas indicadas pelo orientador e professores, por pesquisas realizadas na biblioteca ou através da internet e a aplicação foi desenvolvida utilizando a técnica de prototipação.

Inicialmente, foi realizada uma pesquisa sobre web design e algoritmos genéticos para a estruturação e definição do escopo do algoritmo. Após essa fase, foi determinada a linguagem e ferramentas em que a aplicação foi desenvolvida.

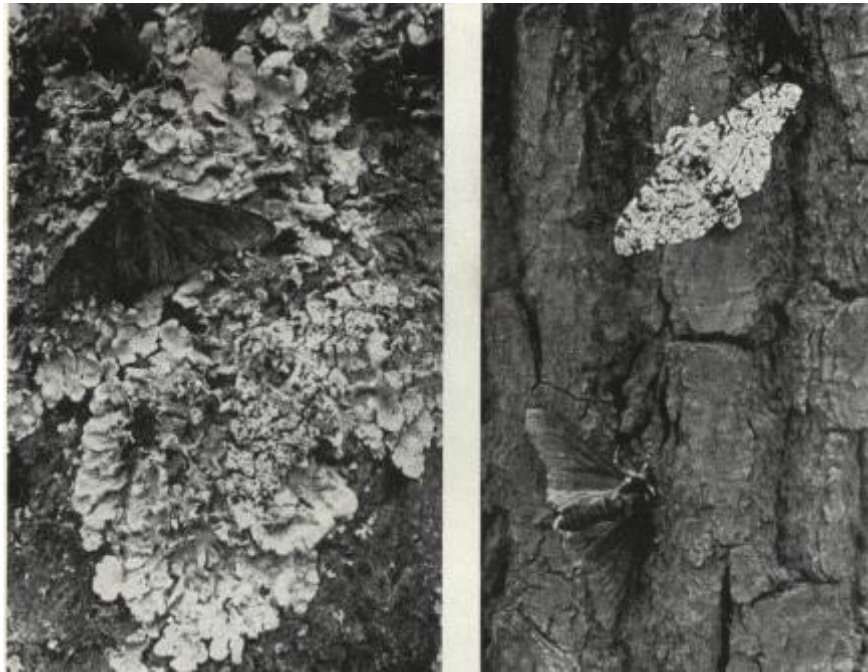
A equipe para este projeto consistiu em: um orientador e um programador. Os materiais utilizados foram: computadores e um ambiente de desenvolvimento para a implementação do algoritmo.

2 REFERENCIAL TÉORICO

2.1 A EVOLUÇÃO

Em 1859 o inglês Charles Darwin publicou uma teoria que abalou a comunidade científica na época e que ainda tem repercussões enormes na atualidade. No seu livro, “A Origem das Espécies”, Darwin teorizou a ideia básica de evolução que serve de base até hoje para os cientistas das mais diversas áreas, “Quanto melhor um indivíduo se adaptar ao seu meio ambiente, maior será sua chance de sobreviver e gerar descendentes”. Darwin, em sua viagem a bordo do navio *HMS Beagle* observou pequenas diferenças em animais da mesma espécie e como essas diferenças poderiam afetar cada indivíduo na luta pela sobrevivência, o que o levou a concluir que alguns indivíduos têm vantagem em algum dos aspectos da sobrevivência, como por exemplo, busca por alimento, sobrevivência através de camuflagem, busca de um parceiro para a reprodução e etc. Por serem diferenciados de uma maneira favorável, o ambiente, no qual estes indivíduos estão inseridos, seleciona aqueles que estão melhor adaptados (Darwin,1859). Com isso, o cientista explicava a origem das espécies e que, teoricamente, todos os indivíduos partiram de um animal em comum, mas ele não conseguiu explicar por que esses indivíduos tinham essas pequenas diferenças entre si e portanto a teoria da origem das espécies estava incompleta. Na atualidade, a teoria melhor aceita, na comunidade científica, é a do neodarwinismo que incorpora à teoria original de Darwin os conceitos de genética como a mutação e a recombinação dos genes, que finalmente explicam as diferenças entre indivíduos da mesma espécie, como por exemplo, a pigmentação das mariposas da Figura 1.

Figura 1– Diversidade de mariposas



Fonte: <http://evolucionismo.org/>

2.2 A VARIAÇÃO

Antes da teoria de Darwin, os naturalistas achavam completamente problemática a inconsistência das características encontradas em indivíduos que compartilhavam a mesma espécie (Wallace, 1889). Segundo Darwin (1859) quando comparados dois indivíduos da mesma espécie sempre haverá alguma diferença, seja ela física ou comportamental, e são essas singularidades que permitem com que as espécies se garantam no meio em que estão inseridas. As variações de espécie podem ser qualquer característica, seja ela física, instintiva ou comportamental, e podem trazer tanto benefícios como malefícios (Darwin, 1859). Segundo Wollaston (1856) é possível observar em indivíduos órgãos de variação, os quais normalmente se diferenciam entre dois membros da mesma espécie. O cientista cita alguns destes órgãos em insetos que podem ser tomados como exemplo. Dentre os atributos que variam, Wollaston (1856) cita: tamanho das asas, frequência da batida de asas, pigmentação do inseto (observado na figura 2), tamanho dos segmentos abdominais e etc.

Figura 2– Diversidade em joaninhas



Fonte: <http://readingevolution.com/evolution.html>

2.3 A SELEÇÃO NATURAL

Na natureza, todos os organismos sofrem com perigos das mais diversas vertentes durante toda a sua existência e sua única chance de sobrevivência são seus atributos genéticos. As propriedades iniciais de um indivíduo, e isso incluem seus hábitos e instintos, são suas únicas ferramentas na luta pela sobrevivência em um ambiente hostil. Devido à variabilidade entre cada organismo dentro de uma mesma espécie, alguns indivíduos se adaptam melhor às condições impostas pelo ambiente em que residem, com isso, o organismo melhor acomodado tem uma vantagem adaptativa sobre o outro e isso aumenta suas chances de vencer a luta pela sobrevivência, reproduzir e, portanto, passar suas características vencedoras para uma próxima geração. Um exemplo deste processo é observado na figura 3, onde os besouros que tem pigmentação verde são facilmente percebidos pelos pássaros predadores, diferentemente dos insetos de coloração marrom que ficam melhor camuflados, o que faz com que eles não sejam um alvo fácil da predação. Logo, pode-se perceber um aumento de besouros marrons quando comparado aos verdes, o que significa que o inseto marrom está melhor adaptado a este ambiente e foi selecionado para passar seus genes. Essa luta pela sobrevivência escolhendo, dentre

uma geração inteira, apenas os indivíduos mais adaptados foi o que Darwin chamou de seleção natural (Wallace, 1889).

Figura 3- Seleção natural dos besouros



Fonte: http://ecologia.ib.usp.br/evosite/evo101/images/micro_mech_4.gif

2.4 A GENÉTICA

A genética viria a explicar uma das falhas da teoria de Darwin. Em seu livro, Darwin cita que a prole dos indivíduos selecionados pelo ambiente herda as características que tornaram seu gerador um vencedor na seleção natural, entretanto, o cientista falha em discursar a respeito de como esse mecanismo funciona, deixando uma missão para os próximos cientistas (Darwin, 1859). Em 1865, os primeiros avanços na área da genética biológica são alcançados, neste ano o pai da genética, Gregor Mendel (figura 4), publica seu trabalho “Experimentos na hibridização de plantas” onde escreve sobre seus diversos experimentos com a ervilha-de-cheiro (*Pisum sativum*).

Figura 4- Gregor Mendel



Fonte: http://upload.wikimedia.org/wikipedia/commons/3/3d/Gregor_Mendel_oval.jpg

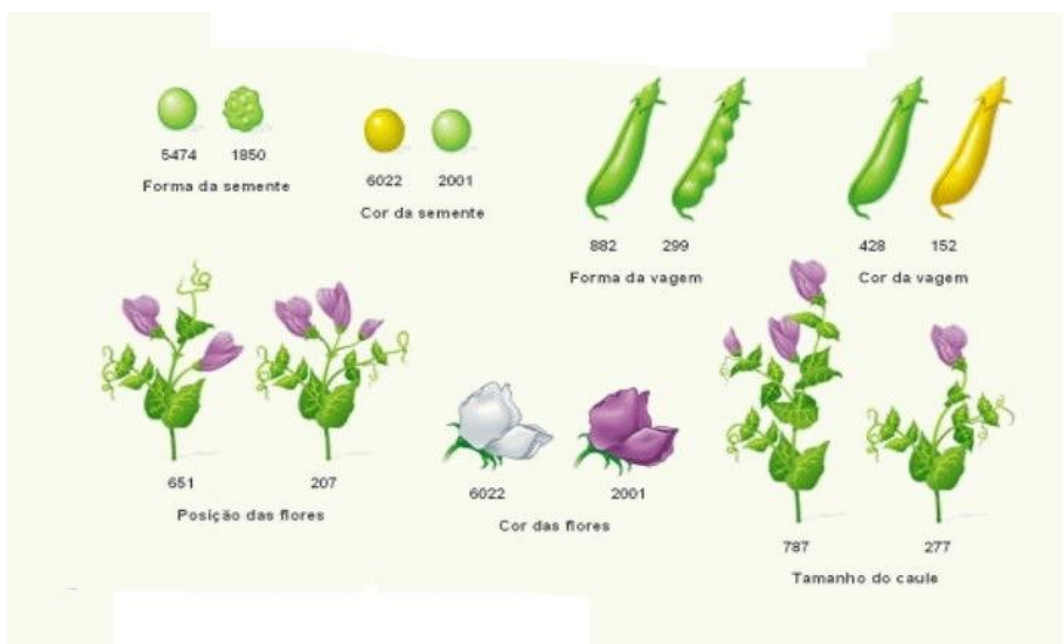
2.4.1 O EXPERIMENTO DE MENDEL

Gregor Mendel teve grande destaque na área da hereditariedade e hoje é conhecido como o pai da genética. Em seu experimento, Mendel escolheu como alvo de pesquisa a planta ervilha-de-cheiro (*Pisum sativum*) devido, primeiramente, à essas ervilhas serem amplamente disponíveis nos vendedores de semente em sua região em uma ampla variedade de formas e cores, fato que facilitou a identificação e a análise. Segundo, por essas plantas possuírem características que possibilitam, tanto a autofecundação, quanto a polinização cruzada (Griffiths et al, 2013). Essas plantas se autofecundam pois as partes masculinas (anteras) e femininas (ovários) ficam envolvidas por duas pétalas fundidas, formando um compartimento chamado de carena, portanto o pesquisador pode remover as anteras de uma planta e fazer a polinização cruzada ou escolher em não remover as anteras e a planta se autofecundará (Griffiths et al, 2013).

O cientista, então, escolheu características chaves das plantas (observadas na figura 5), as quais ele iria observar e cultivou por dois anos várias linhagens para certificar-se de que eram linhagens puras. Uma linhagem pura é uma população que não apresenta variação no caráter observado, isto é, toda a prole produzida por autofecundação ou cruzamento dentro da população apresenta essa característica inalterada. Com isso, Mendel teve uma base extremamente sólida para fazer

experimentos com a hereditariedade, pois qualquer planta de uma população distinta provocaria uma mudança na prole e, portanto, tendo a probabilidade de alterar a característica estudada (Griffiths et al, 2013). Um dos atributos que serviu de experimento para Mendel, foi com a cor da flor das ervilhas, onde uma das linhagens tinha uma flor de cor purpura e a outra de cor branca. O pesquisador então fez cruzamentos entre as flores purpuras e brancas e percebeu que sempre a prole resultava em purpura, não importando como foi feita a fecundação, o que o levou a concluir que a planta purpura era mais “forte”. Em uma nova etapa do procedimento, o pesquisador deixou com que a prole gerada se auto reproduzisse, como resultado surgiram algumas poucas plantas brancas o que levou Mendel a concluir que as plantas purpuras puras quando misturadas com brancas puras geravam uma prole purpura com potencial para gerar crias brancas. Esses seriam os primeiros passos para a definição do que viria a ser um gene (Griffiths et al, 2013).

Figura 5- Fenótipos da ervilha-de-cheiro



Fonte: <http://www.educadores.diaadia.pr.gov.br/arquivos/File/tvmultimedia/imagens/2010/biologia/9caracteres.jpg>

2.4.2 O GENE

Mendel, após observar os resultados de seus diversos experimentos, chegou em uma definição básica de gene. Observando os resultados das flores, onde

linhagens puras de flores brancas e purpuras, quando cruzadas, geravam flores purpura, mas quando essa prole gerada se auto reproduzia resultavam em uma prole com algumas flores brancas, levou Mendel a crer que existem para cada característica de um indivíduo, duas unidades básicas que são determinantes da mesma, uma dominante e uma recessiva. Esses determinantes de hereditariedade foram denominados genes. Mendel também contou o número de indivíduos de cada fenótipo (característica) e se deparou com uma proporção fenotípica de 3:1 (observada na figura 6) onde para cada quatro indivíduos gerados, três teriam a característica dominante e um teria a característica recessiva em cruzamentos entre duas linhagens puras distintas ou entre linhagens impuras. Ele também observou que a única explicação lógica para que a prole pudesse ter características recessivas fosse que os pais passassem somente um gene cada para cada característica, assim como funcionam os gametas, ovócitos e espermatozoides e a união de dois genes formaria um novo fenótipo cujas características dependeriam de se seus genes fossem recessivos ou dominantes (Griffiths et al, 2013).

Figura 6- Tabela de cruzamento entre genes

Cruzamento Monohíbrido

		Gâmetas Masculinos	
		S	s
Gâmetas Femininos	S	SS	Ss
	s	Ss	ss


Fenótipos: 3/4 S: 1/4 s

Fonte: <http://wikiciencias.casadasciencias.org/wiki/images/thumb/d/d0/QPunnet2.jpg/400px-QPunnet2.jpg>




















Mendel, na continuação de seus experimentos, descobriu que, estudando dois fenótipos ao mesmo tempo, a proporção de 3:1 não era mantida, mas sim uma

nova proporção de 9:3:3:1 (Figura 7) tomava o seu lugar, isso levou o pesquisador a concluir que cada alelo (formas alternativas do gene) não dependem de alelos de outro gene, o que ficou conhecida como segunda lei de Mendel. Futuramente, William Baterson e R. C. Punnet encontrariam uma falha nas proporções de Mendel e se descobririam os processos de variabilidade genética (Griffiths et al, 2013).

Figura 7- Proporção 9:3:3:1 em labradores



AaBb

 AaBb	 	AB	Ab	aB	ab
AB	 AABB	 AABb	 AaBB	 AaBb	
Ab	 AABb	 AAbb	 AaBb	 Aabb	
aB	 AaBB	 AaBb	 aaBB	 aaBb	
ab	 AaBb	 Aabb	 aaBb	 aabb	

Fonte: <http://4.bp.blogspot.com/-DksIrIzmlg8/Ucobq-hBOpI/AAAAAAAAAgU/bbCpgz2hKE0/s640/cao.png>

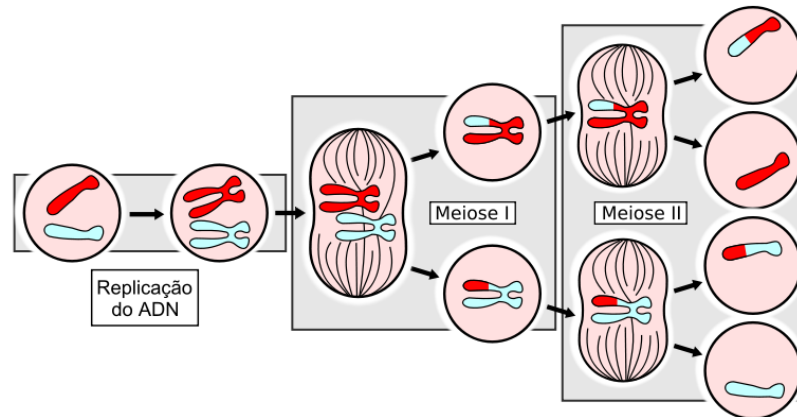
2.4.3 A TEORIA CROMOSSÔMICA DA HERANÇA

A partir da teoria de Mendel, era possível fazer manipulações genéticas com uma boa precisão sem mesmo saber como realmente esses genes eram constituídos biologicamente. Com isso, foi inserido um conceito simples que ficou conhecido como a teoria cromossômica da hereditariedade, essa teoria dita que os genes estão na composição dos cromossomos, o que forçou uma fusão entre a genética e a citologia.

Existem dois tipos de divisões celulares, a mitose e a meiose (Figura 8). A divisão celular realizada na mitose simplesmente copia os cromossomos para uma nova célula, que será exatamente igual a geradora, já na meiose ocorre uma grande variação combinatória chamada de *crossing-over* (Griffiths et al, 2013). Esse

processo da meiose viria a explicar uma grande parte deixada de lado na teoria da evolução.

Figura 8– Meiose



Fonte: <http://rachacuca.com.br/media/educacao/artigo/meiose/esquema-da-meiose-fases-i-e-ii.png>

2.4.4 VARIABILIDADE GENÉTICA

No início do século XX, pesquisadores, ao fazerem experimentos com as ervilhas-de-cheiro, perceberam que os resultados de cruzamentos de dois fenótipos distintos não obedeceram a proporção 9:3:3:1 proposta por Mendel, e então começou uma busca mais profunda para determinar os fatores que geram a variabilidade genética.

2.4.5 MUTAÇÃO

A mutação são mudanças nas sequências de nucleotídeos do material genético de um organismo e podem ser causadas por diversos fatores como radiação ultravioleta ou ionizante, mutagênicos químicos ou até alguns vírus (Griffiths et al, 2013).

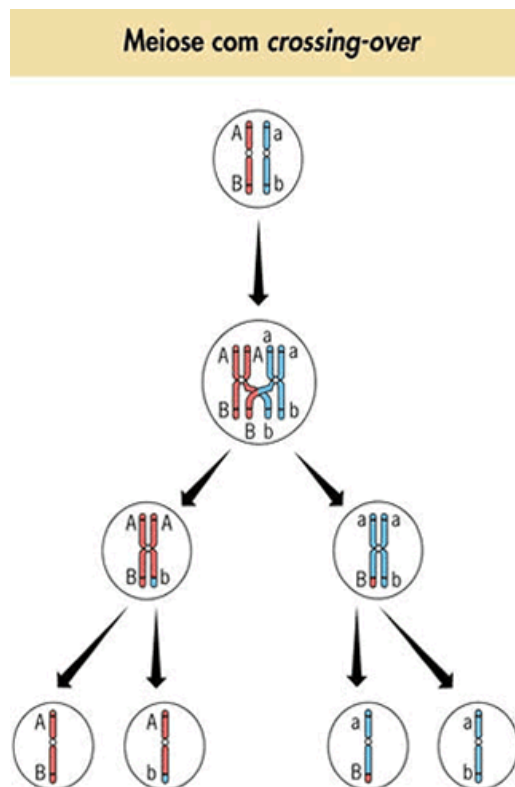
2.4.6 CRUZAMENTO GENÉTICO

O cruzamento genético, ou *crossover* (Figura 9), é um processo que ocorre no procedimento de divisão celular chamado de meiose onde a célula divide seu

código genético para a formação dos gametas para, na fecundação, formar um indivíduo com o número de cromossomos da espécie reestabelecido.

O cruzamento genético ocorre quando duas partes de cromossomo se encostam e acabam por trocar sequências de nucleotídeos, o que gera uma variabilidade que não depende de nenhum fator externo, como na mutação, o que garante a variabilidade genética e, portanto, a evolução das espécies. (Griffiths et al, 2013).

Figura 9- Crossing-over



Fonte: <http://www.sobiologia.com.br/conteudos/figuras/Citologia2/meiose4.JPG>

2.5 ALGORITMOS GENÉTICOS

2.5.1 COMPUTAÇÃO EVOLUTIVA

Computação evolutiva é uma área de pesquisa dentro da ciência da computação que, como o nome sugere, tem como inspiração a teoria da evolução das espécies. Esta metáfora traz uma relação poderosa com a evolução natural para um estilo de resolução de problema muito conhecido, a tentativa-e-erro.

A ideia de aplicar princípios da teoria proposta por Darwin para resolver problemas de forma automatizada vem desde os anos quarenta. Em 1948, Allan Turing propôs uma busca genética ou evolucionária, e em 1962 Bremermann executou experimentos em otimização utilizando evolução, e recombinação e nessa mesma década surgiram diversas vertentes dessa frente de pesquisa como Fogel, Owens e Walsh com a introdução da programação evolutiva (Fogel, 1996) entre muitos outros. No começo dos anos 90 uma nova corrente de pensamentos surgiu com a base na computação evolucionária, como a programação genética proposta por Koza (1992), que gera programas inteiros utilizando os conceitos da evolução e genética. Esse campo de pesquisa acabou por gerar alguns caminhos de pensamento como a programação evolutiva e os algoritmos genéticos.

2.5.2 BREVE HISTÓRICO DOS ALGORITMOS GENÉTICOS

Algoritmos genéticos foram inventados por John Holland (Koza,1992) (Figura 10) nos anos sessenta e foram desenvolvidos por ele, seus estudantes e companheiros na Universidade de Michigan nos Estados Unidos durante os anos sessenta e setenta. A visão inicial de Holland era de criar um algoritmo que, diferente da programação evolucionária que se aplicam em problemas específicos, fosse mais fiel aos fenômenos de adaptação que ocorrem na natureza e como esses fenômenos poderiam ser incorporados nos sistemas computadorizados. Em seu livro de 1975, Holland apresentou o algoritmo genético como uma abstração da evolução biológica, a fim de otimizar uma função pré-determinada. Em seu algoritmo, Holland utiliza uma população de “cromossomos” (uma *string* contendo zeros e uns) onde cada gene é representado por um bit e uma instância de um alelo (0 ou 1), e portanto resultando em um fenótipo diferente dependendo do seu valor. A introdução de conceitos como a população com *crossover*, inversão e mutação foi uma grande revolução na área de computação evolutiva.

Figura 10 - John Henry Holland

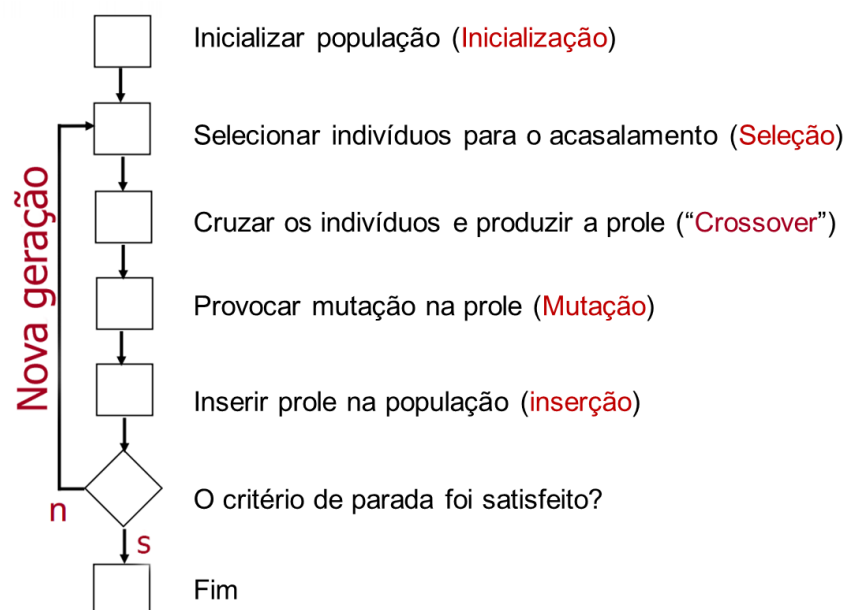


Fonte: <http://um2017.org/faculty-history/sites/default/files/imagecache/small/Holland,%20John%20H.jpg>

2.5.3 O ALGORITMO GENÉTICO

O Algoritmo genético (A.G.) não tem uma forma padrão, mas a comunidade científica da área dita que todos os algoritmos genéticos devem ter os passos observados na figura 11 (Mitchell,1998):

Figura 11- Algoritmo genético

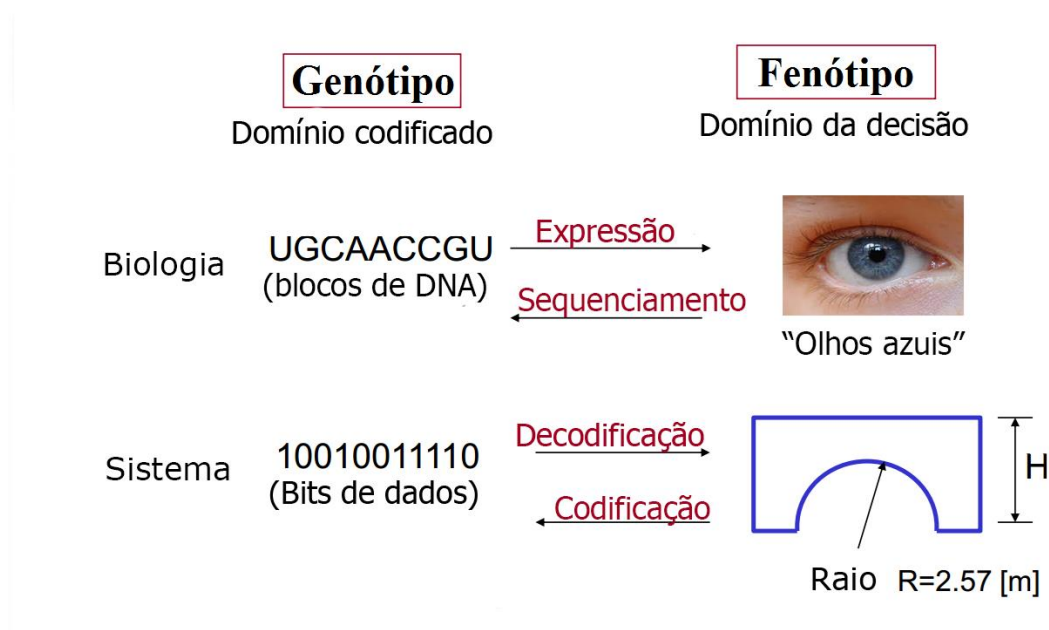


Fonte: Elaborada pelo autor (baseado em: <http://www.ele.ita.br/~flavios/listex2.gif>)

2.5.4 O CROMOSSOMO

O cromossomo é uma das partes mais fundamentais de um algoritmo genético. O AG clássico tem seus indivíduos como cadeias de caracteres com zeros e uns, representando um número (Figura 12). Este número seria o fenótipo do cromossomo gerado, ou seja, a cadeia de bits quando decodificada formaria um número, que representa uma solução provável ao problema.

Figura 13- Tradução genótipo para fenótipo na biologia e em algoritmos genéticos



Fonte: Elaborada pelo autor

O cromossomo pode provocar grande impacto no sucesso do algoritmo genético e não necessariamente deve ser uma cadeia de bits, ele deve ser adaptado ao problema (Linden,2008).

2.5.5 INICIALIZAÇÃO

O primeiro passo de um algoritmo genético é a sua inicialização, onde são gerados os primeiros cromossomos de todo algoritmo. Segundo Tanomaru (1995), a população inicial deve ser gerada através de algum processo heurístico, pois sem variedade não há como aplicar a seleção natural. Segundo Linden (2008), existem

algumas maneiras de tentar gerar uma população inicial melhor qualificada, e assim acelerar o processo de otimização. Entre esses métodos estão:

1. Gerar indivíduos completamente diferentes, assegurando que não haja repetições.
2. Gerar indivíduos divididos igualmente no espaço para que haja uma melhor distribuição de valores iniciais.

Na maioria dos casos, essas técnicas não são aplicadas, devido ao pequeno, se não nulo, custo-benefício resultante.

2.5.6 SELEÇÃO

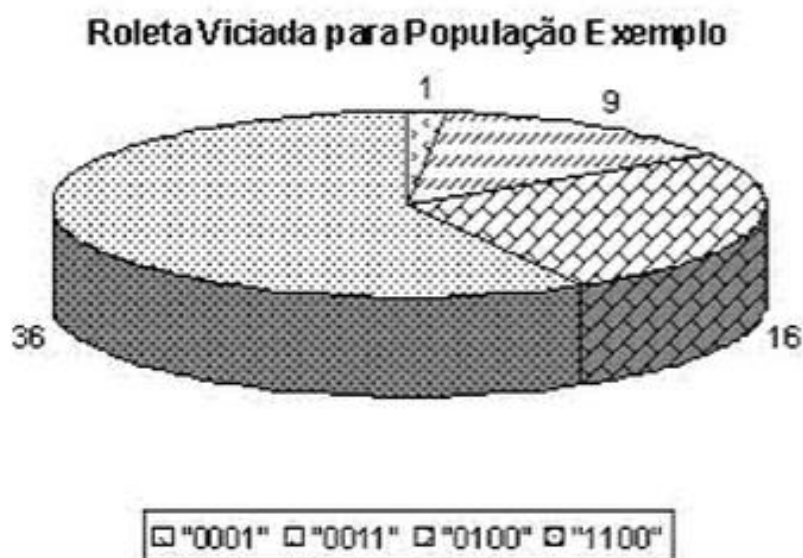
A seleção é a etapa dos algoritmos genéticos em que são selecionados os cromossomos que representam os melhores resultados para o problema. Essa avaliação é feita através de uma função de aptidão¹ (também conhecida como função *fitness*) (Mitchell,1998). A função de aptidão é a ligação fundamental do algoritmo genético com o problema a ser otimizado. Por exemplo, se um AG tem como seu objetivo maximizar a função, então essa própria função seria a função *fitness* e portanto, os maiores valores da função terão uma maior nota de aptidão (Linden, 2008). A partir das notas obtidas da função *fitness* serão selecionados os pais cujo as informações serão transmitidas para as novas gerações, entretanto, ter uma boa nota não significa que o indivíduo será selecionado. A razão para tal fenômeno é a de que um cromossomo que não teve uma boa nota pode ter características interessantes para uma melhor resolução do problema (Mitchell,1998). Segundo Linden (2008), se não houvesse essa chance de utilizar alguns indivíduos que não tem uma nota tão boa a população iria conter indivíduos cada vez mais parecidos, o que geraria uma menor variabilidade genética podendo, assim, afetar a evolução dos resultados. Este efeito é denominado convergência genética.

¹ função de aptidão (ou função *fitness*): é a função a ser otimizada, e é utilizada no algoritmo genético para avaliar os indivíduos, e funciona como se fosse o ambiente, selecionando os membros mais adaptados de cada geração.

Existem alguns métodos de seleção que garantem a convergência e também mantêm alguns dos resultados não tão interessantes para uma melhor variabilidade (Mitchell,1998). São eles:

1. Roleta viciada: este método é o mais utilizado por pesquisadores (Linden,2008). Consiste em criar uma roleta virtual em que cada cromossomo recebe uma faixa de números da roleta proporcional à sua avaliação, isso faz com que os melhores indivíduos de uma população tenham maior probabilidade de serem escolhidos, mas não exclui os que tem uma nota pior (Figura 13), como um exemplo, se uma população tem 3 indivíduos com notas 10, 20 e 30, respectivamente, o primeiro indivíduo receberá uma faixa com dez números (de 1 à 10), o segundo indivíduo receberá uma faixa com vinte números (de 11 à 30) e o terceiro indivíduo receberá uma faixa com trinta números (de 31 à 60), se é então sorteado um número entre um e o somatório das notas e o indivíduo que conter a faixa que contém esse número será selecionado.

Figura 14 - Roleta viciada



Fonte: <http://www.algoritmosgeneticos.com.br/Figura04.03.jpg>

2. Escalonamento Sigma: segundo Mitchell (1998) pesquisadores de AGs fizeram experimentos com vários métodos de escalonamento, métodos que mapeiam valores de aptidão para valores esperados, para fazer com que o AG esteja menos suscetível a uma convergência prematura. Um exemplo de escalonamento sigma pode ser visto na Figura 14.

Figura 15- Escalonamento sigma

$$\text{ExpVal}(i, t) = \begin{cases} 1 + \frac{f(i) - \bar{f}(t)}{2\sigma(t)} & \text{if } \sigma(t) \neq 0 \\ 1.0 & \text{if } \sigma(t) = 0, \end{cases}$$

Fonte: “MITCHELL, 1998, p.168”

Onde $\text{ExpVal}(i, t)$ é o valor esperado de um indivíduo i no tempo t , $f(i)$ é a aptidão média da população no tempo t e $\sigma(t)$ é o desvio padrão da aptidão da população no tempo.

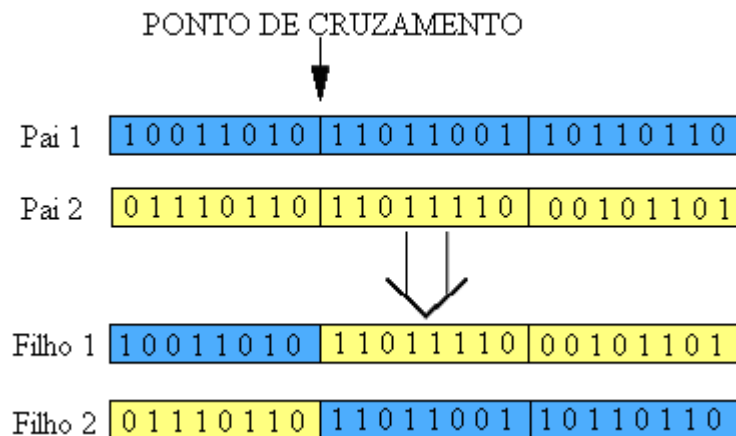
4. Seleção por torneio: é um método muito utilizado, devido a simplicidade e eficiência. Na seleção por torneio existe um parâmetro denominado tamanho do torneio(K). Este parâmetro define quantos indivíduos serão selecionados de forma aleatória na população para competir, e então dentre as duplas dos selecionados, o mais apto é escolhido (de Pinho et al, 2013).
5. Seleção elitista: não é propriamente um método de seleção, mas sim um artifício utilizado para ajudar a garantir a convergência do algoritmo. Quando utilizado, o elitismo faz com que o melhor apto sempre seja mantido na próxima geração (Linden,2008).

2.5.7 CROSSOVER

O crossover (ou *crossing over*) é um operador do algoritmo genético responsável pelo progresso em direção à solução do problema. Como dito na seção 6.3, na Biologia, o crossover é um procedimento que acontece na meiose e é um dos grandes responsáveis pela variabilidade genética nos seres vivos. Na etapa da prófase-I da meiose, dois cromossomos ficam emparelhados e trocam uma porção distal do seu DNA. Nesse processo, é gerado um gene completamente diferente, com características de ambos os DNAs geradores. Qualquer par de cromossomos homólogos pode sofrer crossover três ou quatro vezes durante a meiose. Isto ajuda a evolução ao aumentar a diversidade e diminuir a ligação hereditária entre genes do

mesmo cromossomo. No algoritmo genético clássico, o *crossover* é feito no âmbito binário, onde parte dos indivíduos é mesclada fazendo com que sua prole seja uma mistura de ambos os selecionados, a fim de criar uma geração mais forte do que a anterior, mas com menor variabilidade (Holland,1965).

Figura 16– Exemplo de Crossover em um AG



Fonte: <http://rived.mec.gov.br/atividades/biologia/externos/AGPM/AGPMaula1.htm> (Última visita: 13/05/14)

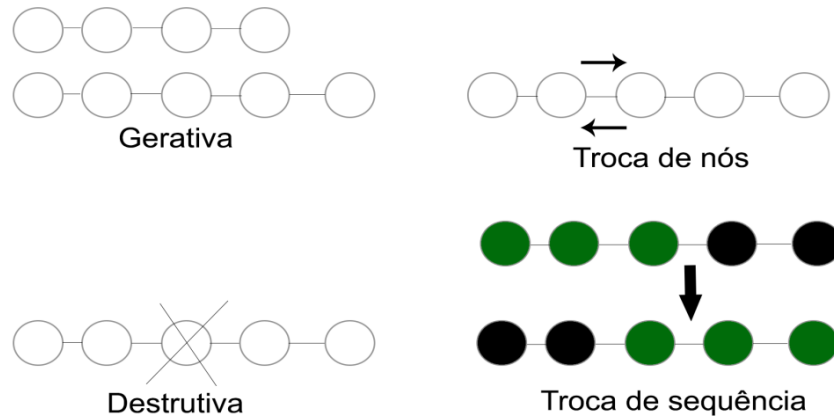
Como se pode observar na figura 15, os selecionados tem um ponto de cruzamento no seu genoma, que pode ser variável, possibilitando uma geração de grande diversidade através de seus membros antepassados. O ponto de corte (ou ponto de cruzamento) constitui numa posição entre dois genes de um cromossomo. Cada indivíduo de n genes contém $n-1$ pontos de corte, e cada ponto de corte será a posição onde as partes dos cromossomos serão trocadas. Esse ponto de corte é sorteado e então os dois pais selecionados trocam de genes. Um algoritmo genético pode utilizar vários pontos de cruzamento na etapa de *crossover* (Linden,2008).

2.5.8 MUTAÇÃO

Segundo Mitchell (1998), uma visão comum na comunidade dos algoritmos genéticos desde a época dos livros de Holland, tende a ter uma visão em que o operador genético mais importante é o *crossover* e é o que gera uma maior variabilidade e inovação, e a mutação assegura que a população não fique fixada em uma direção de uma só solução o que ajuda a prevenir o algoritmo de entrar em um vale com soluções não tão boas. Holland (1992) descreveu as mutações em seu livro como uma chance de ocorrerem alterações nos cromossomos gerados pela recombinação dos cromossomos selecionados. Segundo Tanomaru (1995), a mutação pode ocorrer, entre outras, das seguintes formas (veja figura 16):

1. Gerativa: é gerada uma nova característica no cromossomo, adicionando uma propriedade extra ao resultado.
2. Destrutiva: uma das características pré-existentes do cromossomo é descartada, resultando em um cromossomo de tamanho menor e com menos propriedades.
3. Troca de nós: características do cromossomo trocam de posição, não alterando o tamanho, mas alterando o fenótipo gerado pelo cromossomo.
4. Troca de sequência: essa mutação faz com que segmentos do cromossomo troque de posição, fazendo com que o fenótipo gerado seja profundamente alterado.

Figura 17– Mutações

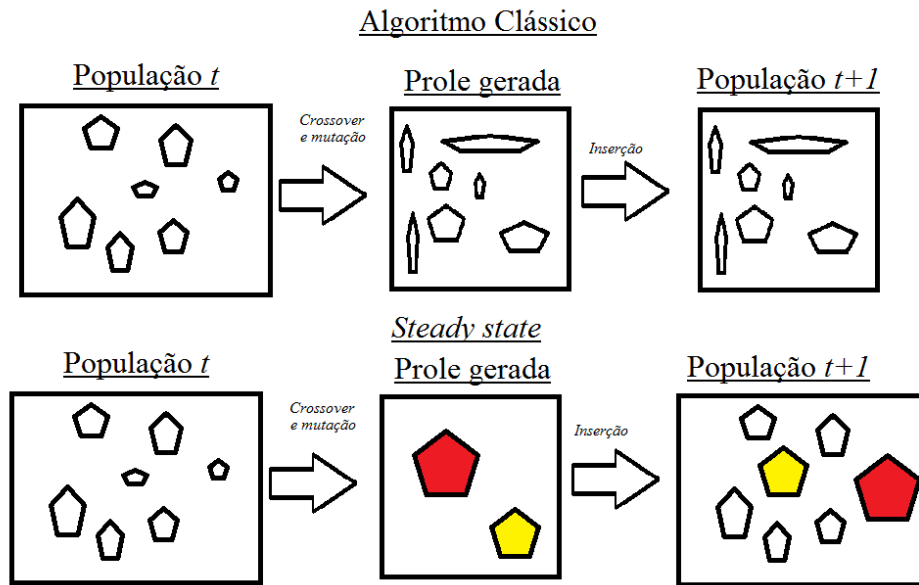


Fonte: Elaborada pelo autor

2.5.9 INSERÇÃO

A inserção é a fase onde os cromossomos gerados e possivelmente mutados são inseridos no algoritmo novamente (Holland, 1975). Em seu livro, Holland (1992) descreveu a inserção como uma troca de populações, onde a antiga é completamente removida e a nova é inserida para fazer parte de uma nova iteração executada pelo algoritmo, mas, segundo Linden (2008), no “mundo real” essa troca é mais suave, sendo que quando uma nova geração é introduzida, ela ainda convive com a anterior, portanto, “isto quer dizer que indivíduos da geração $t+1$ podem procriar com indivíduos da geração t ” (LINDEN, 2008, p. 148). O nome desta técnica é *steady state* (figura 17), e a ideia por trás dela é reproduzir exatamente este tipo de característica das populações biológicas, onde ao invés de serem geradas novas populações inteiras, são gerados “filhos” de um a um ou de dois a dois, e na fase da inserção somente o pior pai (ou piores pais) serão substituídos pela prole gerada. Como nessa técnica preserva-se sempre os melhores resultados e troca-se sempre os piores por resultados melhores, a convergência normalmente é mais rápida, porém, existe uma grande perda de diversidade, o que pode fazer com que o algoritmo sofra com uma convergência genética prematura.

Figura 18– Inserção Steady-state



Fonte: Elaborada pelo auto

2.5.10 ALGORITMOS GENÉTICOS INTERATIVOS

Algoritmos genéticos interativos (AGIs) são uma versão estendida dos algoritmos genéticos onde a avaliação dos indivíduos é realizada pelo usuário. Isto significa que o usuário pode tanto dar uma nota para os indivíduos, e estas notas serem usadas como resultados de uma função de aptidão, quanto escolher os indivíduos que irão dar à luz a nova população. Em ambos os casos, o usuário pode controlar a pesquisa genética como desejar, por exemplo, de acordo com suas preferências estéticas (Olive, Monmarché e Venturini, 2002). O primeiro AGI foi proposto por Dawkins em um de seus livros (Dawkins, 1986) para dar destaque ao poder da seleção natural e da acumulação de pequenas mudanças em um processo evolutivo.

2.6 A WEB

A *World Wide Web* é uma rede que evolui constantemente e que já progrediu muito além das expectativas em sua concepção do início dos anos noventa, quando foi criada para resolver um problema específico: A transferência da enorme quantidade informações dos experimentos realizados no CERN (Laboratório

Europeu para Física de Partículas). Cientistas ao redor do mundo não conseguiam ter o acesso dos importantes avanços científicos que o laboratório estava obtendo. Na época, a Internet já existia, com algumas centenas de milhares de computadores conectados a ela, e foi assim que Tim Berners-Lee (um cientista do CERN) desenvolveu um método de acessar esta rede utilizando um *framework* de *hyperlinks*, que ficou conhecido como *HyperText Transfer Protocol*, ou HTTP. Ele também criou a linguagem *HyperText Markup Language*, ou HTML. Para unir essas duas ferramentas, ele escreveu o primeiro navegador da web e o primeiro servidor web, o que hoje é tomado por mundano, mas na época foi um conceito que revolucionou a transferência de dados em longas distância e deu um grande passo em direção a globalização mundial criando conexões entre diversos pontos do planeta (Nixon,2014).

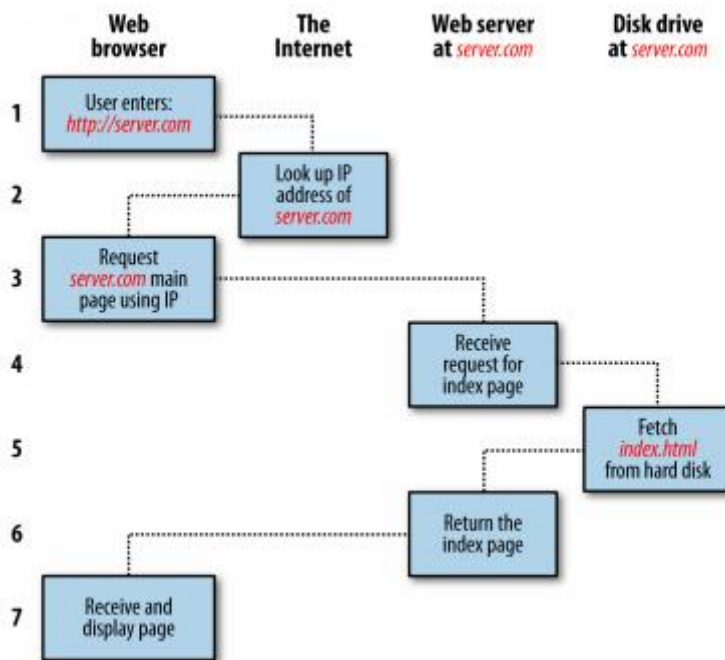
2.6.1 O HTTP

HTTP é o protocolo de comunicação padrão que regulamenta requisições e respostas que acontecem entre o servidor e navegador web. O trabalho do servidor é aceitar as requisições de um navegador cliente e tentar responder a elas de uma maneira correta, normalmente respondendo as páginas requeridas. A outra parte deste processo é o cliente, que no caso, seria um navegador. Entre essa relação de servidor-cliente, podem existir outros diversos participantes, como *proxies*, *gateways*, roteadores e etc. O propósito desses participantes extras é de garantir que a transferência de requisitos e respostas ocorram sem problemas. Um servidor web normalmente tem a capacidade de tratar múltiplas requisições simultaneamente e, quando não está em contato com um cliente, está verificando se algum cliente está tentando fazer novas conexões. Quando uma conexão é feita, o servidor envia de volta uma resposta que a confirma.

2.6.2 O PROCESSO DE REQUISIÇÃO/RESPOSTA

O processo de requisição/resposta no seu nível mais baixo (figura 18) consiste no navegador web pedindo para o servidor enviar uma página e o servidor enviando o site requerido de volta. O navegador então mostra a página.

Figura 19– Processo de requisição/resposta simples



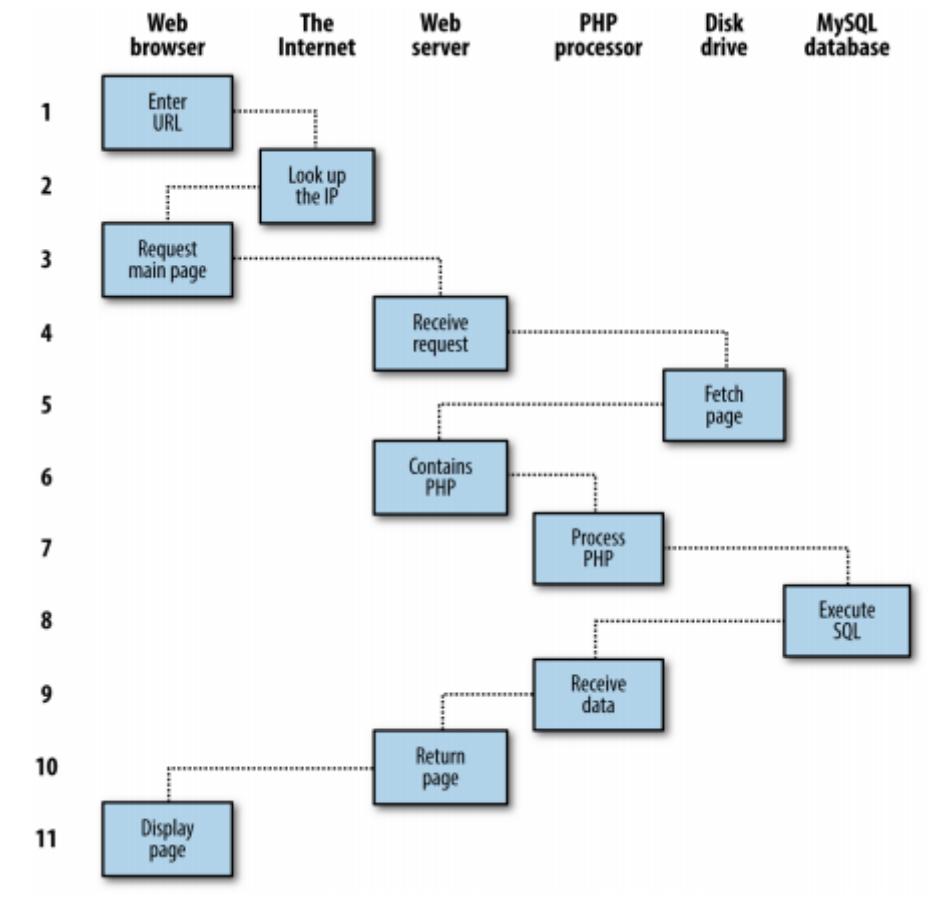
Fonte: “DAVIS; PHILLIPS, 2007, p.3”

Cada passo no processo segue a sequência:

1. O usuário entra com o endereço HTTP no servidor, por exemplo, *http://server.com*.
2. O navegador do usuário procura pelo endereço de IP correspondente ao site, no caso, *server.com*.
3. O navegador, então, envia uma requisição para a *home-page* do site, no caso, *server.com*.
4. A requisição atravessa a Internet e chega no servidor web do site.
5. O servidor web, tendo recebido a requisição, procura pela página no disco que corresponde ao pedido.
6. A página é encontrada pelo servidor e enviada como resposta ao navegador cliente.
7. O navegador do usuário, então, mostra a página com todas as suas características.

Para páginas dinâmicas o processo é um pouco mais extenso (figura 19), pois envolve mais interações com o servidor:

Figura 20-Processo de requisição/resposta dinâmico



Fonte: “DAVIS; PHILLIPS, 2007, p.4”

1. O usuário entra com o endereço HTTP no servidor, por exemplo, *http://server.com*.
2. O navegador do usuário procura pelo endereço de IP correspondente ao site, no caso, *server.com*.
3. O navegador, então, envia uma requisição para a *home page* do site, no caso, *server.com*.
4. A requisição atravessa a Internet e chega no servidor web do site.
5. O servidor web, tendo recebido a requisição, procura pela página no disco que corresponde ao pedido.
6. Com a *home page* na memória, o servidor web percebe que a página tem algum script de alguma linguagem *server-side*, como o PHP que será utilizado nesse exemplo, e passa a página para o interpretador PHP.
7. O interpretado PHP executa o código PHP.

8. Se o código PHP conter alguma conexão com o banco de dados e a página tiver alguma requisição ao banco de dados, por exemplo MySQL, o interpretador passa para a *engine* de banco de dados.
9. O banco de dados retorna os resultados para o interpretador PHP.
10. O interpretador PHP retorna os resultados da execução do código PHP, junto com os resultados do banco de dados para o servidor web.
11. O servidor web retorna a página para o cliente que fez o pedido, onde é mostrada no navegador.

2.7 APLICAÇÃO WEB

Uma aplicação web é qualquer aplicativo que usa um navegador web como um cliente. A aplicação web tira a responsabilidade do desenvolvedor de construir um cliente para um tipo de computador ou sistema operacional específico. Como o cliente é executado em um navegador da Web, o usuário poderia utilizar quase qualquer dispositivo para utilizar a aplicação. Aplicativos da Web geralmente usam a combinação de *scripts* do lado do servidor (ASP, PHP, etc.) e de *scripts* do lado do cliente (HTML, Javascript, etc.) para desenvolver o aplicativo. O script do lado do cliente lida com a apresentação da informação, enquanto o script do lado do servidor lida com toda a parte de armazenamento, recuperação e processamento da informação (Nations, 2011).

3 MATERIAIS

3.1 HTML

É uma linguagem de marcação utilizada para produzir páginas na Web. Documentos HTML podem ser interpretados por navegadores. Segue um exemplo da estrutura básica de um HTML na figura 20.

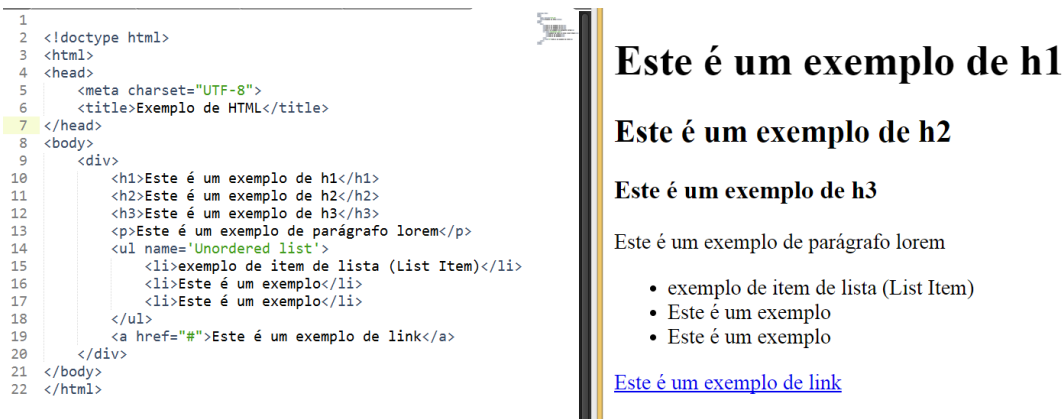
Figura 21- Estrutura básica de um código HTML3

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Document</title>
6 </head>
7 <body>
8
9 </body>
10 </html>
```

Fonte: Elaborada pelo autor

O código em HTML é composto por *tags* que servem para especificar o que está dentro da mesma.

Figura 22- exemplo de HTML



Fonte: Elaborada pelo autor

Como é possível observar na figura 21, existem algumas *tags* básicas como, por exemplo: *HTML*, *head*, *body*, *div*, *h1*, *h2*, *h3*, *p*, *ul*, *li* e *a*, todas entre os sinais “<” e “>” e para determinar quando são fechadas recebem uma barra (“/”) acompanhada do nome. A *tag* “<html>” indica o começo de um documento html, a *tag* “<head>” tem em seu conteúdo informações importantes para o *browser* como o título da imagem e os tipos de caracteres que haverão na mesma, “<body>” é a *tag* em que o *site* realmente fica, as *tags* “<div>” podem ser vistas como caixas que armazenarão conteúdo e as outras citadas são apenas para estruturação, lembrando que o html forma essencialmente um esqueleto do *website* e o css altera o design (W3C,2006).

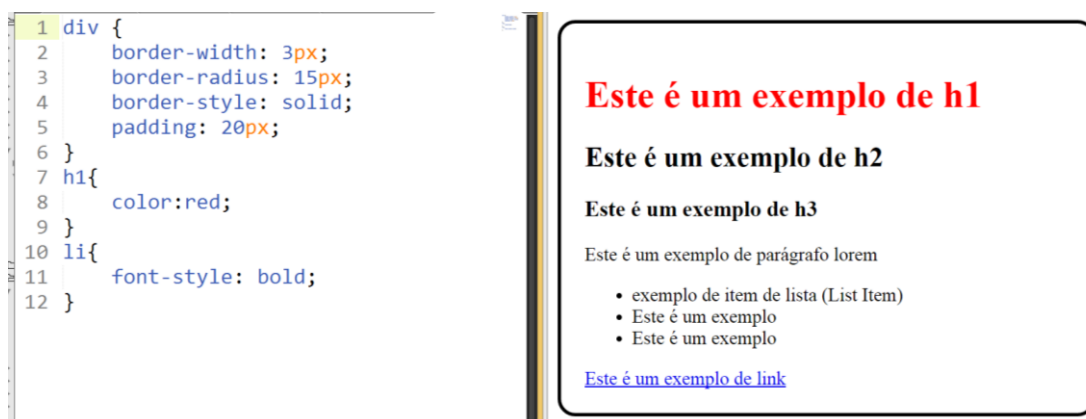
3.2 CSS

É uma linguagem de folhas de estilo utilizada para definir a apresentação de documentos escritos em uma linguagem de marcação, como HTML. Seu principal benefício é prover a separação entre o formato e o conteúdo de um documento (figura 22).

Em vez de colocar a formatação dentro do documento, o desenvolvedor cria um *link* (ligação que fica na *tag* “<head>” do HTML) para uma página que contém os estilos. O código do CSS consiste em customizar *tags* ou usar ids ou classes para identificar o que deseja ser customizado.

Com a variação de atualizações dos navegadores, o suporte ao CSS pode variar. O Internet Explorer² 6, por exemplo, tem suporte total a CSS1 e praticamente nulo a CSS2. Navegadores mais modernos como Google Chrome e Mozilla Firefox tem suporte maior, inclusive até a CSS3, ainda em desenvolvimento (W3C, 2006).

Figura 23 - exemplo de uma página com CSS



Fonte: Elaborada pelo autor

3.3 O PHP

O PHP é uma linguagem de programação geralmente utilizada como *server-side scripting language* e foi criada para desenvolvimento de páginas da web dinâmicas. Essa linguagem oferece suporte integrado para bancos de dados, como o MySQL, o que o torna um ótimo candidato para aplicações na web, desde simples páginas pessoais até páginas complexas de ambientes empresariais. O PHP não tem nenhuma de suas partes processadas pelo navegador, mas sim pela máquina que serve o documento, ou seja, o servidor web. Com o PHP, é possível criar páginas da web dinâmicas, ou seja, páginas que mudam de acordo com condições. Por exemplo: quando um usuário se conecta em uma conta de uma rede social, este usuário vê o conteúdo correspondente a suas informações, portanto este usuário está carregando dados de uma fonte que poderia ser acessada por outro, mas o servidor iria responder com informações correspondentes apenas para aquele usuário. Isto seria impossível

² Internet Explorer: Navegador web que permite você acesse a Internet, veja vídeos, escute musica, jogue e interaja com documentos virtuais da internet, também conhecidos como páginas da web.

utilizando apenas documentos HTML, porque eles são estáticos, e portanto, imutáveis. Todo usuário iria visualizar o mesmo conteúdo da página HTML. O PHP é, também, uma linguagem interpretada, o que traz uma grande vantagem para os desenvolvedores, já que não é necessário compilar o código e, portanto, existe grande facilidade de alteração e teste de um documento [Lengstorf; Hansen, 2014].

3.4 FRAMEWORKS

Para a realização da aplicação desenvolvida neste trabalho foram utilizados alguns *frameworks*. Um *framework* de *software* é uma arquitetura reutilizável que fornece uma estrutura genérica e um comportamento para uma família de abstrações de *software*. *Framework* de *software* são também chamados de *Frameworks* de aplicação (Riehle, 2000). Um *framework* não é uma aplicação completa: normalmente tem “espaços” a serem preenchidos pelo programador, a fim de adequar-se ao problema proposto, portanto, um *framework* fornece a estrutura e alguns mecanismos que interagem entre seus componentes seguindo uma regra estabelecida pelo programador para criar uma aplicação. (Riehle, 2000).

3.5 LARAVEL

O Laravel é um *framework* para aplicações web, e foi escolhido para facilitar o desenvolvimento do algoritmo na parte *backend* da aplicação. Ele tem uma comunidade muito ativa e uma documentação *online* de fácil entendimento, o que facilita o aprendizado e aperfeiçoamento do programador. Este *framework* foi escolhido, também, pela facilidade de se criar moldes para a parte de *frontend* (HTML e CSS) com a *engine* de *templates* embutida, o Blade. Essa *engine* facilita a troca de componentes visuais de maneira dinâmica e reduz a sintaxe necessária para fazer algumas operações e comandos muito utilizados na programação web, como os comandos *echo*, *for* e *foreach*. Outro motivo da escolha do *framework*, foi a possibilidade de expansão do projeto, já que o Laravel possui características muito interessante a projetos de médio a grande porte, como estruturação em *MVC*³,

³ *MVC* é um padrão de arquitetura de software que separa a informação (e as suas regras de negócio) da interface com a qual o usuário interage.

adequação para o uso de protocolos REST, facilidade de conexão com a maioria das *engines* de banco de dados e etc.

3.6 TWITTER BOOTSTRAP

O Bootstrap é um *framework* de web voltado para o desenvolvimento de design. Criado por uma das grandes marcas da web, o Twitter, este *framework* utiliza de regras de CSS pré-definidas que estilizam páginas da web de uma maneira elegante, tornando muito mais fácil o trabalho de um *web designer*. Entre as vantagens do Bootstrap estão:

1. Ele é “*mobile first*”: isto significa que este *framework* tem como alvo principal os dispositivos mobile, que são mais difíceis de trabalhar devido ao seu tamanho diminuto, e portanto facilita com que o design seja funcional em diversas plataformas sem grande alterações no código.
2. Têm diversos recursos: o Bootstrap vem com diversos recursos embutidos, como carrossel de imagens, botões, ícones e etc. Essa foi uma das características fundamentais para a escolha desse *framework*.
3. O “*Grid System*”: é uma técnica de posicionamento muito popular no desenvolvimento de design de páginas na web, que consiste em dividir a página em colunas (no caso do *framework* escolhido são doze) e designar a cada elemento um tamanho baseado nessas colunas. Este foi, também, um fator muito importante na decisão da escolha desse *framework*, já que a posição e tamanho são características muito importantes no design e poder manipulá-las facilmente é muito interessante para o algoritmo genético.
4. Comunidade ativa: o Bootstrap é muito utilizado por web designers, e por isso existe uma grande quantidade de componentes não nativos do *framework* que podem ser livremente utilizados, esse fato aumenta muito a quantidade de alelos que podem ser utilizados

4 METODOLOGIA

4.1 O ALGORITMO DESENVOLVIDO

4.2 DEFINIÇÃO DO CROMOSSOMO

Como visto na fundamentação teórica, o cromossomo é uma peça chave de um algoritmo genético bem sucedido. Para o trabalho foram feitas duas abordagens para definir o cromossomo. O primeiro pensamento foi tomar cada alelo (pedaço de um cromossomo) como uma linha de código (HTML para a parte estrutural e CSS para a parte de estilização) e um cromossomo seria composto de uma cadeia de linhas de código para formar uma estrutura completamente randômica. Esse processo tem como vantagem uma grande variabilidade, devido ao grande número de combinações de código possíveis, mas experimentos rápidos confirmaram que existe um grande problema com esse tipo de cromossomo: a convergência. Nos experimentos, o algoritmo não conseguiu atingir uma estrutura básica para uma página na web. Para resolver este problema, foram utilizados alelos mais complexos, componentes bem estruturados, como imagens, carrosséis de imagem, estruturas com texto e imagem e etc. Além disso, os indivíduos foram divididos em *head*, *body* e *footer*, onde cada um desses segmentos tem informações mais específicas para as devidas posições, onde por exemplo, no segmento *head* o algoritmo utiliza componentes como barras de navegação, que costumam ficar em uma parte superior na maioria dos *sites*. Com isso, o problema de convergência foi amenizado, conseguindo resultados plausíveis com 20 iterações. Mas houve uma perda de variabilidade, o que pode diminuir a eficácia do algoritmo. Estas estruturas são guardadas dentro de vetores multidimensionais, ou seja, uma população é um vetor de sites, onde cada site é um vetor e cada posição do mesmo é uma linha que contém as estruturas mencionadas.

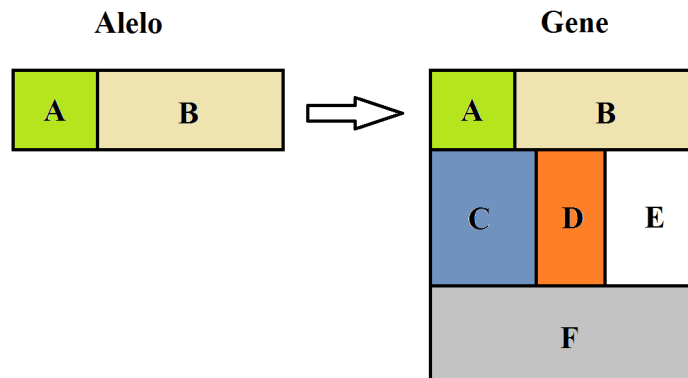
4.3 OS ALELOS

Cada alelo do cromossomo, será uma linha que irá conter componentes de estrutura previamente definida (Figura 24). Esses elementos dependerão do segmento em que serão inseridos, se, por exemplo, o segmento da página em questão for o head, os elementos serão correspondentes a parte superior do site como se pode observar na lista a seguir:

1. Texto.
2. Imagem.
3. Texto com imagem.
4. “Thumbnails”.
5. “Jumbotron” (Componente do Bootstrap para dar destaque a um texto importante).
6. “Slider”
7. “List Group” (Componete do Bootstrap com um grupo de pequenas imagens e textos em forma de lista).
8. “Media object”.

Cada elemento será encapsulado em uma classe do Bootstrap (classe que determina o tamanho de um elemento utilizando o *grid-system*) que determinará o seu comprimento dentro da linha, e sua altura será randomizada entre um máximo e um mínimo, que será determinado pelo usuário antes do início do algoritmo.

Figura 25- O gene da aplicação



Fonte: elaborada pelo autor

4.4 A INICIALIZAÇÃO

O algoritmo genético precisa de uma população inicial, para que se dê início as etapas de seleção. A primeira população do algoritmo é gerada randomicamente, mas segue alguns parâmetros passados pelo usuário na tela inicial, como o tamanho da população, a faixa de altura que um elemento pode ter e o número de linhas do site, onde cada linha tem seus elementos escolhidos de maneira aleatória.

4.5 A SELEÇÃO

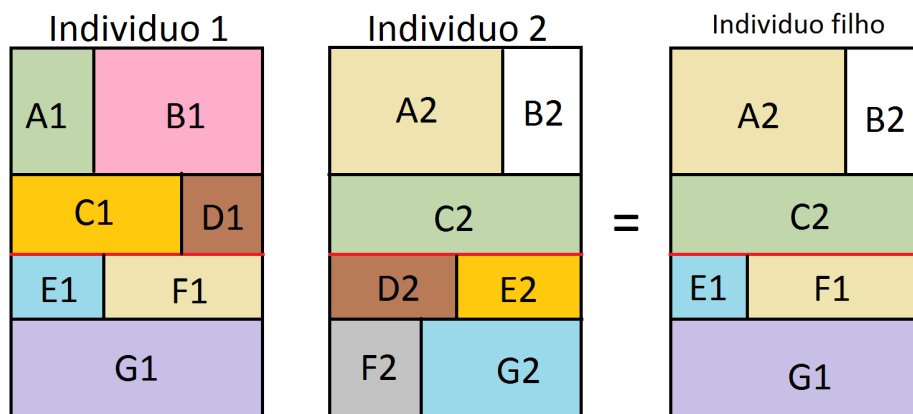
A seleção é feita de maneira interativa, ou seja, o usuário terá grande participação nessa etapa, onde a função de seleção (*fitness*) será desempenhada pelo gosto do usuário. Cada site têm um espaço reservado a sua pontuação onde o usuário irá escolher uma nota para a página (entre 0 a 10 estrelas) que será usada como seu valor de aptidão nos algoritmos de seleção. Com as notas da população já atribuídas, o algoritmo utiliza do método da roleta viciada para selecionar os indivíduos que irão participar do crossover. Neste método são atribuídos faixas de números onde a nota de um site será o comprimento desta faixa, por exemplo, se uma população tem 3 sites, e as notas dos indivíduos desta população são respectivamente: três, cinco e sete, o primeiro site vai ficar com a faixa de números (0,1,2), o segundo ficará com (3,4,5,6,7) e o ultimo com (8,9,10,11,12,13,14). Em seguida, são sorteados 2 números randomicamente, esses números são então comparados com as faixas de

números atribuídas aos indivíduos, e a faixa que tiver algum destes números sorteados será escolhida para o processo de crossover. Vale ressaltar que neste algoritmo é possível um tipo de "reprodução assexuada" onde um indivíduo faz crossover com ele mesmo.

4.6 O CROSSOVER

O crossover é realizado a partir de dois indivíduos, onde esses dois elementos estão estruturados em vetores. Como no algoritmo de Holland(1965), um número aleatório é escolhido, e ele servirá como ponto de corte entre esses dois vetores, e irá definir onde serão feitas as trocas de alelos. Com o valor do ponto de corte é gerado dois indivíduos que serão inseridos na nova população, um levando as posições de um dos pais antes do corte e as posições do outro pai depois do corte e vice-versa (Figura 26). Lembrando que no caso de reprodução assexuada esse processo vai gerar dois clones do indivíduo pai.

Figura 27 - O *crossover* da aplicação



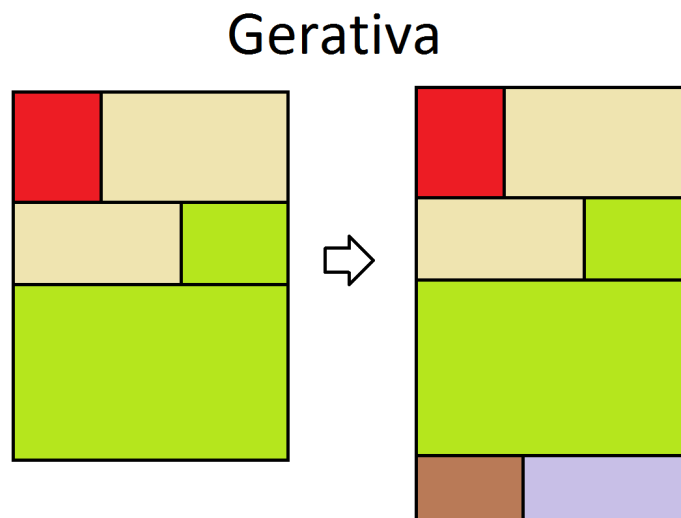
Fonte: elaborada pelo autor

4.7 A MUTAÇÃO

Com os indivíduos da nova população gerados, a etapa de crossover é finalizada com o processo de mutação. Esse processo pode ou não ocorrer, e sua frequência é determinada por uma taxa atribuída pelo usuário, onde quando maior, mais mutações ocorrerão. A ocorrência de mutações é muito importante, pois traz variabilidade genética ao algoritmo mas, se muito elevada, traz problemas de convergência, devido a diversidade muito grande. No algoritmo em questão, a mutação segue os seguintes padrões conforme as classificações de Tanomaru (1995):

1. Gerativa: um novo bloco é gerado com elementos randômicos, ou seja, uma nova posição do vetor é adicionada, gerando uma nova linha no site com elementos novos completamente aleatórios (Figura 28).

Figura 29 - Mutação gerativa

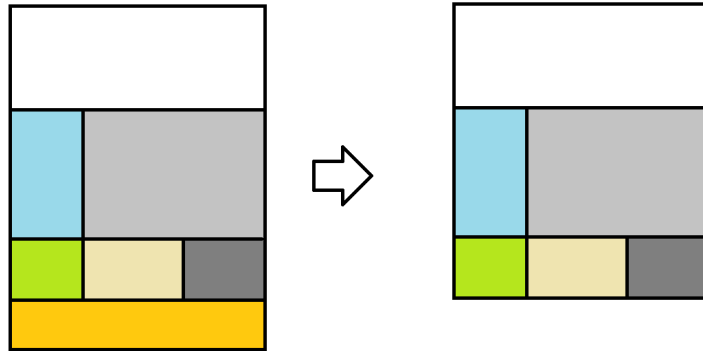


Fonte: elaborada pelo autor

2. Destrutiva: uma linha do site é destruída, e portanto, todas as outras linhas são rearranjadas para se adequarem as novas posições (Figura 30).

Figura 31 - Mutação destrutiva

Destrutiva

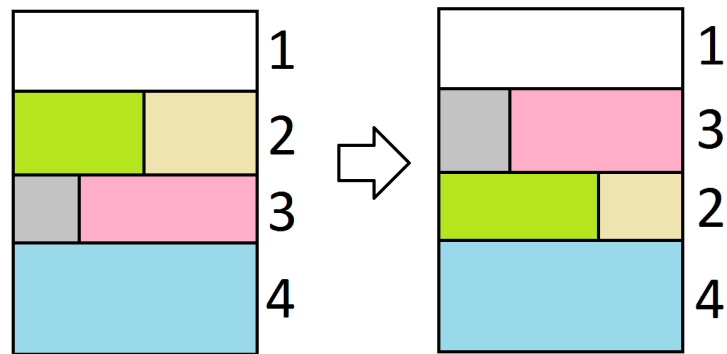


Fonte: elaborada pelo autor

3. Troca de nós: Uma linha do site troca de posição com outra, ou seja, conteúdos de duas posições são trocadas entre si (Figura 32).

Figura 33 - Mutação de troca de nós

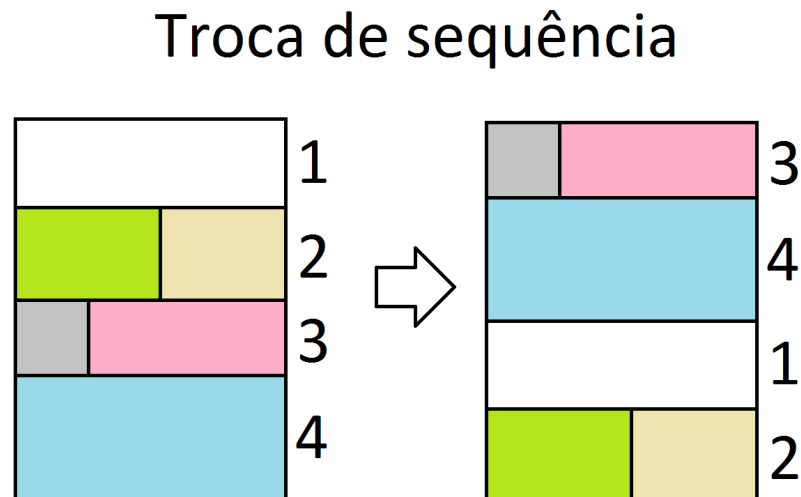
Troca de nós



Fonte: elaborada pelo autor

4. Troca de sequência: neste tipo de mutação é sorteado um número que irá determinar o ponto de troca. As posições anteriores ao ponto são trocadas com as posições posteriores (Figura 34).

Figura 35 - Mutação de troca de sequência



Fonte: elaborada pelo autor

4.8 A INSERÇÃO

Como foi mencionado na secção 4.3 o algoritmo inicial tinha grandes problemas de convergência, já que a variabilidade inicial era demasiadamente grande, portanto foram tomadas algumas providências que ajudaram o algoritmo a convergir para um ponto com mais facilidade, a primeira das ações tomadas foi explicitada na secção 4.3, onde cita que o algoritmo utiliza blocos de componentes bem definidos, a outra providência tomada foi a utilização do método de inserção chamado de elitismo, onde o indivíduo melhor pontuado no algoritmo da seleção é mantido na próxima geração. No caso deste trabalho, o indivíduo de "elite" será o site cujo a nota dada pelo usuário é a maior da população. Apesar de ajudar resolver o problema de convergência, o elitismo traz consigo o problema de diminuir a variabilidade genética, já que preserva características da geração anterior diminuindo, assim, o espaço para novos indivíduos com novas características que poderiam ajudar a encontrar uma melhor solução. Além disso, quando a população é

maior que 50 indivíduos, é utilizada a técnica do *steady state*, mencionada na secção 2.5.9, para aumentar a convergência.

4.9 O CRITÉRIO DE PARADA

Em qualquer algoritmo de otimização uma das etapas mais importantes é o critério de parada, que define quando o processo de otimização será finalizado. Neste trabalho, o critério de parada ficará por conta do usuário, quando estiver satisfeito com o resultado do algoritmo, ou seja, quando um dos indivíduos da população tiver um design agradável a sua preferência estética, ele terá a opção de parar o algoritmo e retornar o site.


4.10 A APLICAÇÃO DESENVOLVIDA

4.10.1 TELA INICIAL

A tela inicial da aplicação (figura 29) contém uma breve explicação da aplicação, bem como links para este trabalho. A tela inicial irá conter um *form* com os parâmetros iniciais do algoritmo:

1. Taxa de mutação: Um número *float* que representará a taxa que irá ditar a probabilidade de ocorrer mutação nos indivíduos produzidos na etapa do *crossover*.
2. Tamanho da população: Um número inteiro que irá representar o tamanho máximo de uma população.
3. Faixa de altura dos elementos: um número *float* mínimo e máximo que irá determinar as alturas dos elemento.
4. Número de linhas do site: Um número inteiro, que irá determinar o número máximo de linhas dos indivíduos.

Figura 36- Tela inicial da aplicação

WebGene  [About Me](#)

Options

Mutation rate	<input type="text" value="10"/>	Population size	<input type="text" value="5"/>
Maximum height	<input type="text" value="500"/>	Minimum height	<input type="text" value="50"/>
Number of lines	<input type="text" value="4"/>	Maximum itens per line	<input type="text" value="3"/>

[Start](#)

Fonte: Elaborada pelo autor

4.10.2 A PRIMEIRA GERAÇÃO

A primeira geração é criada com elementos aleatórios baseados nas opções especificadas pelo usuário na tela de seleção. Esta etapa do algoritmo utiliza de duas funções principais, chamadas de *criar_html()* e *criar_css()*.

Figura 37 - Função criar_html

```
private function (criar_html) ($numero_de_linhas , $max_itens_por_linha , $min_height , $max_height,$css){
    //logica de criação do HTML de um exemplar de site
    $html = array();
    $itens = array();
    $aux_css="";
    $numero_de_linhas=$numero_de_linhas+3;
    foreach ($css as $key => $value) {
        $aux_css.=$value;
    }
    $html[0] = array( 0 =>htmlspecialchars('<!DOCTYPE html><html lang="en">
        <head>
            <link href="css/normalize.css" rel="stylesheet" type="text/css">
            <script src="https://code.jquery.com/jquery-1.11.1.min.js"></script>
            <link href="css/bootstrap.min.css" rel="stylesheet" type="text/css">
            <script src="js/bootstrap.min.js"></script>
            <script src="js/star_rating.js" type="text/javascript"></script>
            <meta charset="UTF-8">
        ');
    ');

    $html[1] = array(0=>htmlspecialchars_decode($aux_css));
    $html[2] = array(0=>htmlspecialchars('
        <title>Webgene</title>
        </head>
        <body>
        '));

    $html[3] = array(0 => htmlspecialchars($this->get_navbar()));
    $html[4] = array(0 => htmlspecialchars('<div class="container">'));
    for ($i=5; $i <= $numero_de_linhas ; $i++) {
        //randomizar altura da linha
        $altura = rand($min_height,$max_height);
        //randomizar numero de itens na linha
        $num_itens = rand(1,$max_itens_por_linha);
        $aux= $num_itens;
        $itens[0] =htmlspecialchars('<div class="row 1_'.($i-1).' ">');
        $tam_item_aux=0;
        $tam_item_total=0;
        for ($j=1; $j <= $num_itens; $j++) {
            //definir tamanho do item e adequar
            $aux--;
            $tam_item = rand(3,12-$tam_item_total-$aux);

            if ($j==$num_itens) {
                $tam_item = 12-$tam_item_total;
            }
            $tam_item_total+=$tam_item;
            $itens[$j] = htmlspecialchars("<div class='col-xs-".$tam_item."'>");
            if ($tam_item>=2) {
                $itens[$j] .= htmlspecialchars($this->get_item());
            }
            $itens[$j] .= htmlspecialchars("</div>");
        }
        $itens[$num_itens+1]=htmlspecialchars('</div>');
        $html[$i] = $itens;
        $itens=array();
    }
    $html[$numero_de_linhas+1] =array(0 => htmlspecialchars('</div>'));
    $html[$numero_de_linhas+2] =array(0 => htmlspecialchars($this->get_footer()));
    $html[$numero_de_linhas+3] = array(0 => htmlspecialchars('</body></html>'));

    return($html);
}
```

Fonte: Elaborada pelo autor

4.10.3 A FUNÇÃO “CRIAR_HTML()”

Na função “*criar_html()*”, observada na figura 30, a *tag head* da página é especificada e é chamada a função que cria o *css*. Após a determinação das informações da *tag head* são determinados os elementos das três principais categorias do *site*: a barra de navegação, o corpo e o *footer* do *site*. A barra de navegação e o *footer* tem funções específicas para a sua criação devido a sua especificidade, e o corpo utiliza a função *get_item()* que escolhe um número randômico e baseado nele, escolhe um elemento para ser utilizado. Dentro da função *criar_html()* também é determinado o tamanho dos elementos dentro de uma linha, onde os tamanhos de cada elemento são randomicamente escolhidos entre o mínimo e máximo disponível pelo *Bootstrap* (de um à doze). Por exemplo: supõe-se que o primeiro tamanho escolhido aleatoriamente seja cinco, o próximo tamanho possível será um número entre um e sete, esse processo se repete até que o tamanho de todos os elementos somados seja doze.

4.10.4 A FUNÇÃO “CRIAR_CSS()”

Nesta função é gerado o código *css* com as cores principais do *site*, para a aplicação serão utilizadas cores tríades⁴ e esta função ira designar uma cor para a barra de navegação, uma cor para o corpo e uma cor para o *footer* (Figura 38).

Figura 39 - Função *criar_css*

```
private function criar_css() {  
    $colors = $this->random_color_triad_calculation();  
    //logica de criação do css do site  
    $css[]=htmlspecialchars('<style>');  
    $css['body']=htmlspecialchars('body {background-color:rgb('.$colors[0][0].','.$colors[0][1].','.$colors[0][2].');}');  
    $css['nav'] =htmlspecialchars('.navbar {background-color:rgb('.$colors[2][0].','.$colors[2][1].','.$colors[2][2].');}');  
    $css['foot']=htmlspecialchars('.footer{background-color:rgb('.$colors[1][0].','.$colors[1][1].','.$colors[1][2].');}');  
    $css[]=htmlspecialchars('</style>');  
    return($css);  
}
```

Fonte: elaborada pelo autor

⁴ cores tríades: esquema que usa 3 cores eqüidistantes na roda de cores.

4.11 TELA DE SELEÇÃO

Nesta tela o usuário irá visualizar os indivíduos da população e então a avaliação será feita através de um botão em um canto superior, onde um *modal* (um container que sobrepõe o site principal e o escurece) irá sobrepor o site e mostrará um sistema de notas baseado em estrelas que podem variar de um à dez e que devem ser selecionadas de acordo com a qualidade estética do site (Figura 40). Esta etapa irá determinar a avaliação de um indivíduo, e por tanto, sua chance de ser selecionado no *crossover*.

Figura 41– Tela de avaliação

A imagem mostra uma interface de usuário para avaliação, intitulada "Avaliar" no topo com um ícone de fechar (X) no canto superior direito. Abaixo do título, há duas linhas de texto: "Geração: 1" e "Indivíduo: 1". No centro da tela, há uma fila de dez estrelas vazias para avaliação. No canto inferior direito, há um botão azul com o texto "Salvar avaliação".

Fonte: elaborada pelo autor

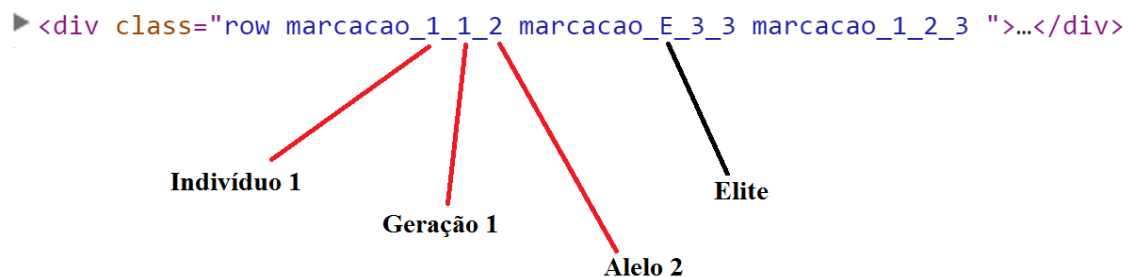
4.11.1 A MARCAÇÃO

A marcação é o fator que indicará para o usuário a origem de cada alelo do algoritmo, bem como suas iterações, o que tornará possível determinar quais partes formaram a pagina observada. Na iteração inicial e na iteração do *crossover* cada componente será marcado seguindo uma regra que irá exibir exatamente o indivíduo, a sua geração e qual é o número de seu alelo, essa marcação será adicionada como

um atributo *class* do HTML. Existem alguns fatores que podem alterar a marcação como por exemplo:

1. Mutação gerativa: Se o alelo foi gerado devido à uma mutação gerativa, o indivíduo terá uma classe HTML denominada “MUTADO”.
2. Mutação de troca de nós ou troca de sequência: Se o alelo foi trocado devido à uma mutação de troca de nós ou troca de sequência, o indivíduo terá uma classe HTML denominada “TROCADO”.
3. Elitismo: Se o alelo participou de uma página de elite, terá uma classe HTML denominada “ELITE”.

Figura 42-Exemplo de marcação



Fonte: Elaborada pelo autor

Como é possível observar no exemplo, na primeira geração do algoritmo, este alelo era parte do indivíduo número um da população e estava na segunda posição do site, na segunda geração este alelo foi utilizado não primeiro indivíduo da nova população e sua posição foi alterada, devido ao *crossover*, da segunda para a terceira, na terceira geração este alelo participou do indivíduo de elite e manteve a posição. O algoritmo na Figura 43 estava em sua terceira geração.

4.11.2 ALGORITMO DE SELEÇÃO

Como mencionado a cima, o algoritmo desenvolvido irá utilizar o método de seleção denominado roleta viciada. Na figura 34 é possível observar o código do algoritmo da roleta viciada.

Figura 44- Código da roleta viciada

```
//algoritmo da roleta viciada

//pegar total de notas
foreach ($sites as $site) {
    $nota_total += $site["nota"];
}
//atribuir pesos da roleta

foreach ($sites as &$amp;site) {
    $site["limite_inferior"] = $limite_inferior;
    $site["limite_superior"] = $limite_inferior + $site["nota"];
    $limite_inferior += $site["nota"];
}
//seleção dos candidatos ao crossover
while (sizeof($cross_pop) < $pop_size) {
    $rand = mt_rand(0, $nota_total);

    foreach ($sites as $site) {

        if ($rand > $site["limite_inferior"] && $rand <= $site["limite_superior"] ) {
            array_push($cross_pop, $site);
            break;
        }
    }
}
//fim
```

Fonte: elaborada pelo autor

Primeiramente é determinado o tamanho da roleta, isso é feito através do somatório de todas as notas atribuídas pelo usuário, depois, é atribuído a cada site a sua faixa de números dentro da roleta, e a seleção é feita a partir de um número gerado aleatoriamente entre zero e o somatório das notas, quando este número randômico se encontra dentro da faixa dada a uma página, este indivíduo é selecionado, e sofrerá *crossover*. Esta população terá o tamanho igual a de todas as populações, que é estipulado na tela inicial.

4.11.3 A FUNÇÃO DE MUTAÇÃO

A função de mutação será chamada para todos os elementos de todos os sites da população resultante do crossover. Como pode-se observar na figura 35, a função só irá alterar o código de um elemento se um número randômico respeitar certo limiar, baseado na taxa de mutação

Figura 45- Função de mutação

```
private function mutacao($site,$rate){  
    $itens = array();  
    if (mt_rand(1,100)<=$rate) {  
  
        //sortear o tipo de mutacao  
        switch (mt_rand(1,4)) {  
  
            case '1'://gerativa  
                break;  
  
            case '2'://destrutiva  
                break;  
  
            case '3'://troca de nós  
                break;  
  
            case '4'://troca de sequencia  
                break;  
  
        }  
  
    }  
    return($site);  
}
```

Fonte: elaborada pelo autor

Se o número aleatório não ultrapassar o limiar, é escolhida aleatoriamente uma das seguintes mutações:

1. Gerativa: se é adicionada uma linha nova utilizando os mesmos conceitos da função *criar_html()*, vale ressaltar que quando é gerada uma linha por essa mutação a classe *MUTATED* será adicionada no *HTML* (vide Figura 36).

Figura 46- Mutação gerativa

```
case '1'://gerativa

    //randomizar altura da linha
    $altura = rand(100,500);
    //randomizar numero de itens na linha
    $itens[0] =htmlspecialchars('<div class="row MUTATED">');
    $num_itens = rand(1,3);
    $aux= $num_itens;
    $tam_item_aux=0;
    $tam_item_total=0;
    for ($j=1; $j <= $num_itens; $j++) {
        //definir tamanho do item e adequar
        $aux--;
        $tam_item = rand(1,12-$tam_item_total-$aux);
        if ($j==$num_itens) {
            $tam_item = rand(12-$tam_item_total,12-$tam_item_total-$aux);
        }
        $tam_item_total+=$tam_item;
        $itens[$j] = htmlspecialchars("<div class='col-xs-".$tam_item." ' >");
        $itens[$j] .= htmlspecialchars($this->get_item());
        $itens[$j] .= htmlspecialchars("</div>");
    }
    $itens[$num_itens+1]=htmlspecialchars('</div>');
    $site["html"][mt_rand(5,count($site["html"])-4)] = $itens;
    //adiciona um item entre a segunda e a penultima posição do site,
    //lembrando que existem espaços reservados!

break;
```

Fonte: elaborada pelo autor

2. Destrutiva: nessa mutação é simplesmente apagada uma linha do vetor que representa o site que sofre a mutação utilizando o comando *unset* do *PHP* (vide Figura 37).

Figura 47 - Mutação destrutiva

```
case '2'://destrutiva

    unset($site["html"][mt_rand(5,count($site["html"])-1)]) ;

break;
```

Fonte: elaborada pelo autor

3. Troca de nós: Dois números aleatórios entre zero e o tamanho do vetor que representa a página são escolhidos para que seja realizada a troca de posição neste vetor. Todos os elementos deste vetor serão marcados com a classe *HTML NODE_SWITCH* (vide Figura 38).

Figura 48 - Mutação de troca de nós

```
case '3'://troca de nós

    $random_node = mt_rand(5,count($site["html"])-4);
    $random_node_2=$random_node;
    while ($random_node_2 == $random_node) {
        $random_node_2 = mt_rand(1,count($site["html"])-4);
    }
    $aux = $site["html"][$random_node];
    $site["html"][$random_node] = $site["html"][$random_node_2];
    $site["html"][$random_node_2] = $aux ;
    foreach ($populacao[$i]['html'] as $key => &$value) {
        $aux = $value[0];
        $pos = strpos($aux, "row");
        if ($pos && ($key!=1 || $key==count($populacao[$i]['html'])-2) ){
            $str_to_insert = "NODE_SWITCH";
            $newstr = substr_replace($aux , $str_to_insert,($pos+4),0);
            $value[0] = $newstr;
        }
    }
break;
```

Fonte: elaborada pelo autor

4. Troca de sequência: se é escolhido um número aleatório entre as possíveis posições do corpo do site e essas posições são trocadas. Todos os elementos deste vetor serão marcados com a classe *HTML SEQ_SWITCH* (vide Figura 39).

Figura 49- Mutaç o de troca de sequ ncia

```
case '4'://troca de sequencia

    $random = mt_rand(5,count($site["html"])-4);
    $aux_head = $site["html"][0];
    $aux_foot = $site["html"][count($site["html"])-4];
    $aux_first_segment = array_slice($site["html"], $random);
    $aux_last_segment = array_slice($site["html"], 0 , $random);
    unset($site["html"]);
    $site["html"]=$aux_head+$aux_last_segment+$aux_first_segment+$aux_foot;
    $pos = strpos($aux, "row");
    foreach ($populacao[$i]['html'] as $key => &$value) {
        $aux = $value[0];
        $pos = strpos($aux, "row");
        if ($pos && ($key!=1 || $key==count($populacao[$i]['html'])-2) ){
            $str_to_insert = "SEQ_SWITCH";
            $newstr = substr_replace($aux , $str_to_insert,($pos+4),0);
            $value[0] = $newstr;
        }
    }
}

break;
```

Fonte: elaborada pelo autor

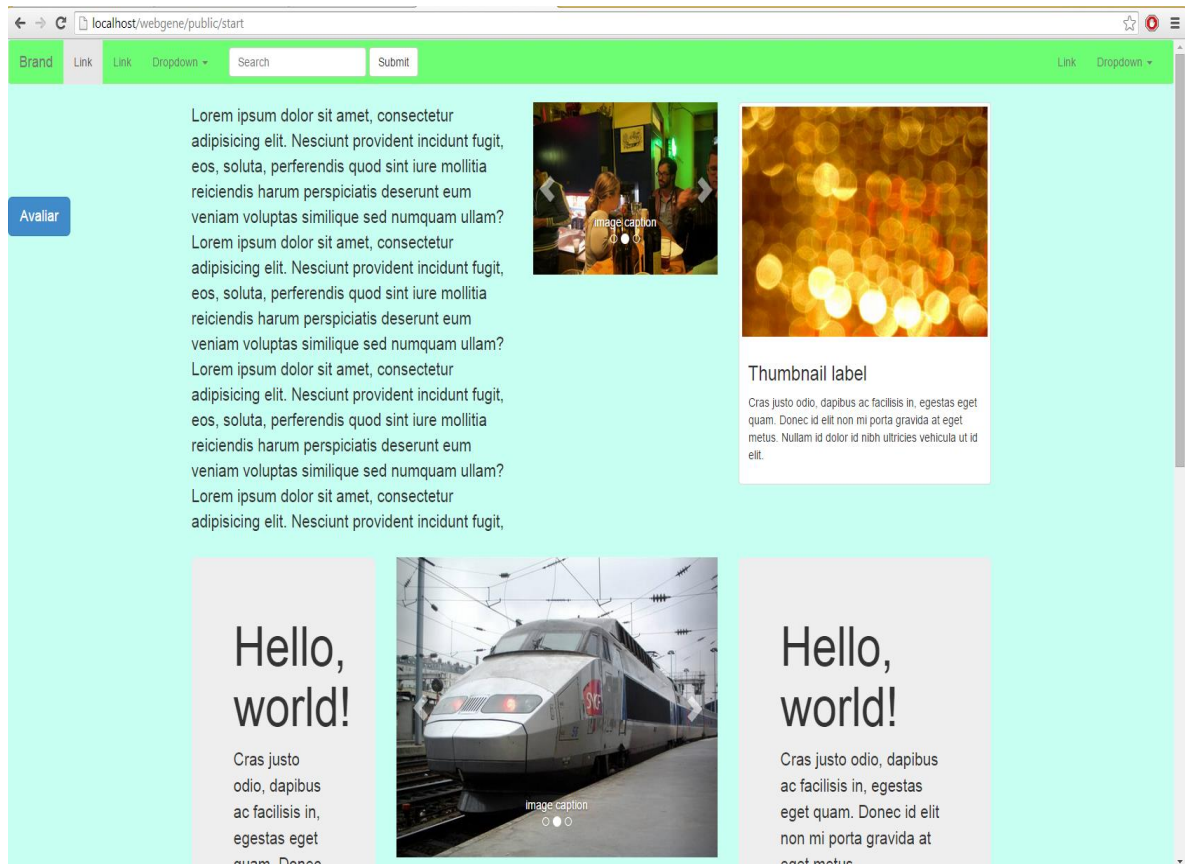
1.1.1. FINALIZA  O

Ap s as etapas explicitadas acima serem executadas, o algoritmo ir  se repetir indefinidamente, mostrando novas popula  es derivadas da primeira popula  o e influenciadas pelas sucessivas sele  es efetuadas pelo usu rio. Cabe ao usu rio finalizar o algoritmo, quando encontrar uma p gina que seja esteticamente agrad vel.

1.1.2. RESULTADOS

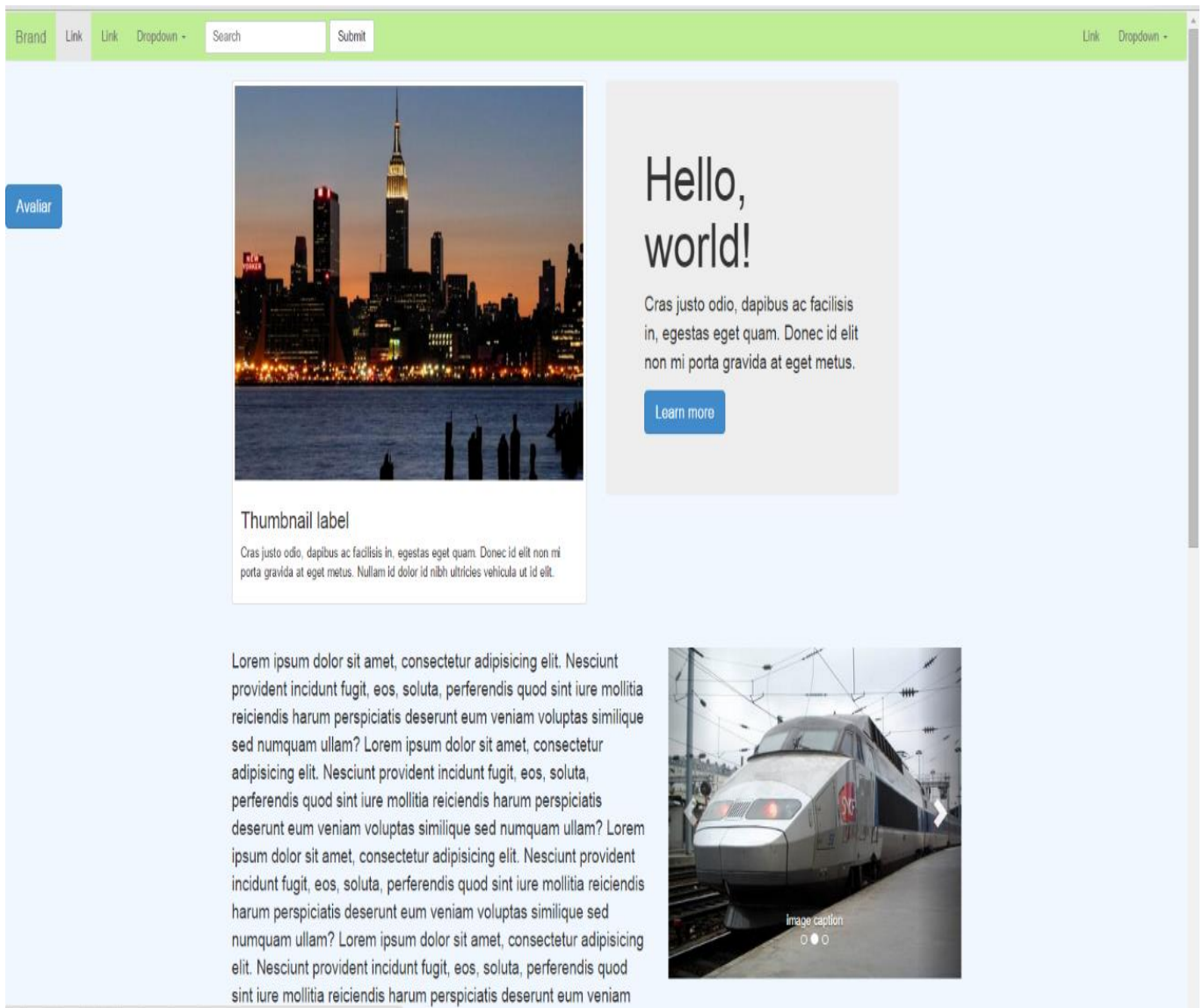
Segue a baixo (figuras 40 e 41, respectivamente) um indiv duo inicial e um resultado obtido ap s 30 itera  es do algoritmo.

Figura 50– Individuo inicial



Fonte: elaborada pelo autor

Figura 51– Indivíduo final



Fonte: elaborada pelo autor

5 CONCLUSÃO

O trabalho tem como objetivo criar uma aplicação que auxilia no design de páginas na web utilizando algoritmos genéticos interativos. A aplicação consegue gerar sites com uma qualidade razoável que servem o propósito de inspirar e dar suporte a decisão do usuário na escolha de um layout e a utilização do algoritmo genético faz com que alguns modelos sejam bem inovadores, sem perder o foco da decisão sempre seguindo a preferência do usuário, o que serve muito bem o propósito do trabalho. Alguns problemas foram encontrados, como a convergência, onde algumas vezes não era possível encontrar um site que fosse agradável, e a falta de variabilidade, onde algumas vezes as populações geradas não tem componentes agradáveis. Mudanças nos parâmetros da aplicação tem grandes resultados com esses dois problemas em particular. É possível concluir com este trabalho que é possível utilizar algoritmos genéticos interativos para auxiliar no design de páginas *web*, servindo de inspiração e como um protótipo para o usuário.

A aplicação desenvolvida obteve um resultado dentro do esperado mas é possível observar grande potencial se houver um refinamento e ampliação do projeto. Alguns dos pontos que podem ser ampliados são: incrementar o número de elementos que possam ser usados como alelos para o algoritmo, aumentando assim, sua variabilidade; possibilitar a adição de um elemento especificado pelo usuário para trazer melhor convergência e melhores resultados e a possibilidade de filtrar quais elementos serão utilizados, para que seja possível uma evolução mais controlada e que consiga atender melhor às necessidades do usuário, como por exemplo, desenvolvimento de páginas com uma função específica, como um formulário de cadastro, por exemplo.

REFERÊNCIAS

BENTLEY, P.J. Generic Evolutionary Design of Solid Objects using a Genetic Algorithm. Division of Computing and Control Systems School of Engineering The University of Huddersfield. 1996.

DARWIN, C. A Origem das Espécies, no meio da seleção natural ou a luta pela existência na natureza, 1 vol., 1859. ISBN:9781481958813

DAVIS, M. E. ;PHILLIPS, J. A. *Learning PHP and MySQL*. O'Reilly Media, Inc., 2006. ISBN: 9780596553500.

DAWKINS, R. *The Blind Watchmaker*. New York: 1986.

FLANDERS, Vincent; PETERS, Dean. *Son of Web Pages That Suck: Learn Good Design by Looking at Bad Design*. Abril, 2002. ISBN: 0-7821-4020-3.

FOGEL, David B. The Advantages of Evolutionary Computation. Natural Selection, Inc.1997

GRIFFITHS, A.J.F.; WESSLER, S.R.; CARROLL, S.B.; DOEBLEY, J. 2013. Introdução à Genética. 6a Edição. Guanabara Koogan. ISBN: 9788527721912

HOLLAND, John H. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Complex Adaptive Systems. Abril 29, 1992. ISBN: 9780262581110.

KOZA, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

LENGSTORF, Jason; HANSEN, Thomas Blom. PHP for absolute beginners. Apress, 2009.

LINDEN, R. *Algoritmos Genéticos*. Brasport, 2008.

LOHN, Jason D.; LINDEN, Derek S.; HORNBY, Gregory S.; KRAUS, William F.; ARROYO, Adán Rodríguez; SEUFERT, Stephen E. Evolutionary Design of an X-Band Antenna for NASA's Space Technology 5 Mission. NASA Ames Research Center. Vol. 3. 2004.

MITCHELL, M. *An Introduction to Genetic Algorithms*. MIT Press, 1998.

NATIONS, D. *What is a Web Application?* .Disponível em <http://webtrends.about.com/od/webapplications/a/web_application.htm>. Acesso em: 23 mar. 2015.

NIXON, R. *Learning PHP, MySQL, and JavaScript: A Step-by-Step Guide to Creating Dynamic Websites*. 2014

OLIVER, Antoine; MONMARCHÉ, Nicolas; VENTURINI, Gilles. Interactive Design of Web Sites with a Genetic Algorithm. In: ICWI. 2002.

DE PINHO, Alexandre F.; MONTEVECHI, José A. B.; MARINS, Fernando A. S.; MIRANDA, Rafael de C. Algoritmos Genéticos: Fundamentos e Aplicações. UDESC Joinville, 2013

RIEHLE, Dirk. *Framework Design: A Role Modeling Approach*, Universität Hamburg, 2000

TANOMARU, Julio. "Motivação, fundamentos e aplicações de algoritmos genéticos." II Congresso Brasileiro de Redes Neurais. 1995.

VEEN, Jeffrey. *The Art & Science of Web Design*. Dezembro, 2000. ISBN: 0-7897-2370-0.

W3C, HTML & CSS. Disponível em: <<http://www.w3.org/standards/webdesign/htmlcss>> . Acesso em: 07 mar. 2015.

WALLACE, A.R. *Darwinism*. Londres: Macmillan, 1889.

WOLLASTON, T. V. *On the variation of species, with especial reference to the Insecta*: followed by an inquiry into the nature of genera. Londres: J. Van Voorst, 1856.