

# Uma Análise das Aplicações dos Algoritmos Genéticos em Sistemas de Acesso à Informação Personalizada

ANTONIO CARLOS C. DA S. SOBRINHO  
MARIA DEL ROSÁRIO GIRARDI

UFMA – Universidade Federal do Maranhão  
Departamento de Pós Graduação em Engenharia Elétrica  
Cx. Postal 99 – CEP 65.000-000 São Luís (MA)  
accss@uol.com.br; rgirardi@deinf.ufma.br

**Resumo:** Uma das principais características da computação evolutiva é a de permitir interpretar processos evolutivos como processos de aprendizado. Populações de indivíduos quando submetidas a essas técnicas de aprendizado, podem ter o seu desempenho verificado por meio de uma função de avaliação específica. Dentro da computação evolutiva, destaca-se o uso dos algoritmos genéticos, que de forma simples, conduzem uma busca, fazendo evoluir um conjunto de estruturas e selecionando de modo iterativo as mais adequadas. Este artigo apresenta uma análise dos algoritmos genéticos e uma arquitetura para a recuperação e filtragem de informação, cuja implementação baseia-se em algoritmos genéticos.

**Palavras Chaves:** Computação evolutiva, Algoritmos genéticos, Recuperação e filtragem de informação.

## 1 Introdução

Há bastante tempo, a natureza têm servido de inspiração ao homem para a criação de máquinas, métodos e técnicas que melhorem sua vida cotidiana. Alguns exemplos típicos desta teoria estão presentes em invenções como: aviões baseados nas características dos pássaros e submarinos com sistemas de imersão semelhante aos dos peixes.

Recentemente novas técnicas têm sido inspiradas na natureza ou na biologia, como as "Redes Neurais", que se baseiam no funcionamento do cérebro humano para prover aos computadores uma chamada "Inteligência Artificial".

Dentro deste contexto, surgia em meados do século XIX um dos mais importantes princípios no campo da evolução da vida, a teoria da Seleção Natural de Darwin [Darwin, 1859], que defendia a idéia de que, na natureza, os seres vivos com melhores características, ou seja, os mais adaptados, tendem a sobreviver frente aos demais.

Baseadas nestes princípios, foram formuladas técnicas de busca, chamadas de Algoritmos Genéticos (AG), para utilização em processos de otimização e resolução de problemas. Estas novas técnicas têm uma vasta aplicabilidade em problemas de Inteligência Artificial, como aprendizagem de máquina, modelagem de usuários e acesso à informação.

Este artigo analisa a utilização de técnicas baseadas em Algoritmos Genéticos em sistemas de acesso a fontes de informação dinâmicas e que necessitem se adaptar aos seus usuários.

Na seção 2 é apresentada a base conceitual dos algoritmos genéticos, descrevendo seus componentes, funcionamento e alguns aspectos de implementação. Na seção 3 é apresentado um estudo de caso, o AMALTHAEA, onde é mostrado como os algoritmos genéticos podem ser usados em sistemas de recuperação e filtragem de informação.

## 2 Algoritmos genéticos

A primeira tentativa de representação, por meio de um modelo matemático, da teoria de Darwin, surgiu com o livro *The Genetic Theory of Natural Selection* [Fisher, 1960]. A evolução era, tal como a aprendizagem, uma forma de adaptação, diferindo apenas na escala de tempo. A evolução, em vez de ser o processo de uma vida, era o processo

de gerações. Como era feita em paralelo por um conjunto de organismos, tornava-se mais poderosa que a aprendizagem.

A seguir, John Holland dedicou-se ao estudo de processos naturais adaptáveis, tendo inventado os algoritmos genéticos em meados da década de 60. Ele desenvolveu os algoritmos genéticos em conjunto com seus alunos e colegas da Universidade de Michigan nos anos 60 e 70, com o objetivo de estudar formalmente o fenômeno da adaptação como ocorre na natureza e desenvolver modelos em que os mecanismos da adaptação natural pudessem ser importados para os sistemas computacionais. Como resultado do seu trabalho, Holland edita *Adaptation in Natural and Artificial Systems* [Holland, 1975] e, em 1989, David Goldberg edita *Genetic Algorithms in Search, Optimization and Machine Learning* [Goldberg, 1989], hoje considerados os livros mais importantes sobre algoritmos genéticos.

Os algoritmos genéticos são técnicas de busca que utilizam procedimentos iterativos que simulam o processo de evolução de uma população de possíveis soluções de um determinado problema. O processo de evolução é aleatório, porém, guiado por um mecanismo de seleção baseado na adaptação de estruturas individuais. A cada iteração do algoritmo (uma geração), um novo conjunto de estruturas é criado através da troca de informações (bits ou blocos) entre estruturas bem adaptadas selecionadas da geração anterior. Novas estruturas são geradas aleatoriamente com uma dada probabilidade e incluídas na população. O resultado tende a ser um aumento da adaptação de indivíduos ao meio, podendo acarretar também em um aumento global da aptidão da população a cada nova geração. Neste caso, a população evolui a cada geração se aproximando de uma solução ótima [Sobrinho, 2003].

## 2.1 Algoritmo genético básico

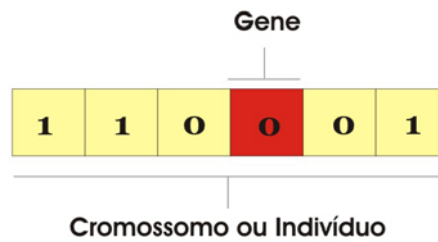
Um algoritmo genético é estruturado de forma que as informações referentes a um determinado sistema possam ser codificadas de maneira análoga aos cromossomos biológicos. Desta forma o algoritmo proposto assimila-se muito ao processo evolutivo natural. O algoritmo genético básico envolve seis passos: codificação das variáveis, criação da população inicial, avaliação da resposta, cruzamento, mutação e seleção dos mais aptos.

O pseudocódigo de um algoritmo genético básico é mostrado na figura 1. Nele podemos ver que os algoritmos genéticos começam com uma população  $P$  de  $n$  estruturas aleatórias (indivíduos), onde cada estrutura codifica uma solução do problema. O desempenho de cada indivíduo é avaliado com base em uma função de avaliação de aptidão. Os melhores tenderão a ser os progenitores da geração seguinte, possibilitando que as suas características sejam transmitidas para as próximas gerações [Palazzo, 1997].

Algoritmo AG	
{ t := 0;	// contador
Inicia_população(P,t);	// iniciar uma população de n indivíduos
Avaliação (P, t);	// avaliar aptidão dos indivíduos da população
Repita até (t = d)	// testar critérios (tempo, aptidão, etc.)
{ t := t + 1;	// incrementar o contador de gerações
Seleção_dos_pais (P,t);	// selecionar os pares para cruzamento
Recombinação (P, t);	// realizar cruzamento dos pares selecionados
Mutação (P, t);	// perturbar o grupo gerado pelo cruzamento
Avaliação (P, t);	// avaliar as novas aptidões
Sobrevivem (P, t)	// selecionar os sobreviventes
}	
}	
onde:	
t – geração atual;	
d – critério para finalizar o algoritmo;	
P – população	

Figura 1: Pseudocódigo básico de um Algoritmo Genético

Na prática, nós podemos implementar facilmente um algoritmo genético com o simples uso de *strings* de bits ou caracteres para representar os cromossomos e, com simples operações de manipulação de bits podemos implementar cruzamento, mutação e outros operadores genéticos. Na figura 2, podemos ver a representação de um cromossomo composto por seis genes através de uma *string* de valores binários.



**Figura 2:** Exemplo de representação de um cromossomo de genes binários

## 2.2 Operadores Genéticos

São os operadores genéticos que transformam a população através de sucessivas gerações, estendendo a busca até chegar a um resultado satisfatório. Um algoritmo genético padrão evolui, em suas sucessivas gerações, mediante o uso de três operadores básicos [Shapiro, 1999]:

- Seleção
- Cruzamento
- Mutação

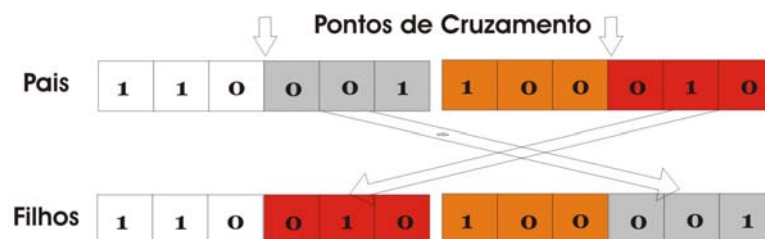
### 2.2.1 Seleção

A idéia principal do operador de seleção em um algoritmo genético é oferecer aos melhores indivíduos da população corrente, preferência para o processo de reprodução, permitindo que estes indivíduos passem as suas características às próximas gerações. Isto funciona como na natureza, onde os indivíduos altamente adaptados ao seu ambiente possuem naturalmente mais oportunidades para reproduzir do que aqueles indivíduos considerados mais fracos.

### 2.2.2 Cruzamento

Uma das principais características dos algoritmos genéticos que os distinguem das demais técnicas de busca é o operador cruzamento [Mitchell, 1996]. Cruzamento é a troca de segmentos entre "casais" de cromossomos selecionados, com a finalidade de originar novos indivíduos que poderão ser incluídos na próxima geração. A idéia central do cruzamento é a propagação das características dos indivíduos mais aptos da população. O operador cruzamento é utilizado após o de seleção. As formas mais comuns de reprodução em algoritmos genéticos são de um ponto de cruzamento, de dois pontos de cruzamento e cruzamento uniforme.

Na reprodução baseada em um ponto de cruzamento (*single-point crossover*), o ponto de quebra do cromossomo é escolhido de forma aleatória sobre a longitude da *string* que o representa e a partir desse ponto se realiza a troca de material cromossômico entre os dois indivíduos. Na figura 3 temos um esquema da representação desse tipo de cruzamento, na qual foi escolhido o ponto de cruzamento 3.



**Figura 3:** Esquema gráfico do cruzamento de um ponto

Na reprodução baseada em dois pontos de cruzamento (*two-point crossover*), procede-se de maneira similar ao cruzamento de um ponto, mas a troca de segmentos é realizada a partir de dois pontos (figura 4).

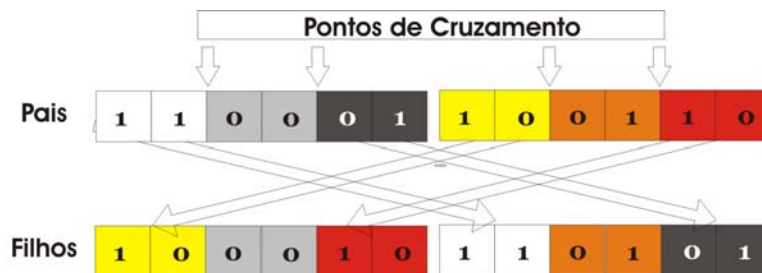


Figura 4: Esquema gráfico do cruzamento de dois pontos.

Na reprodução baseada em cruzamento uniforme (*uniform crossover*), cada gene do descendente é criado através da cópia de um gene dos pais, escolhido de acordo com uma máscara de cruzamento gerada aleatoriamente. Onde houver 1 na máscara de cruzamento, o gene correspondente será copiado do primeiro pai e onde houver 0 será copiado do segundo. O processo é repetido com os pais trocados para produzir o segundo descendente. Uma nova máscara de cruzamento é criada para cada par de pais. A figura 5 demonstra graficamente o processo.

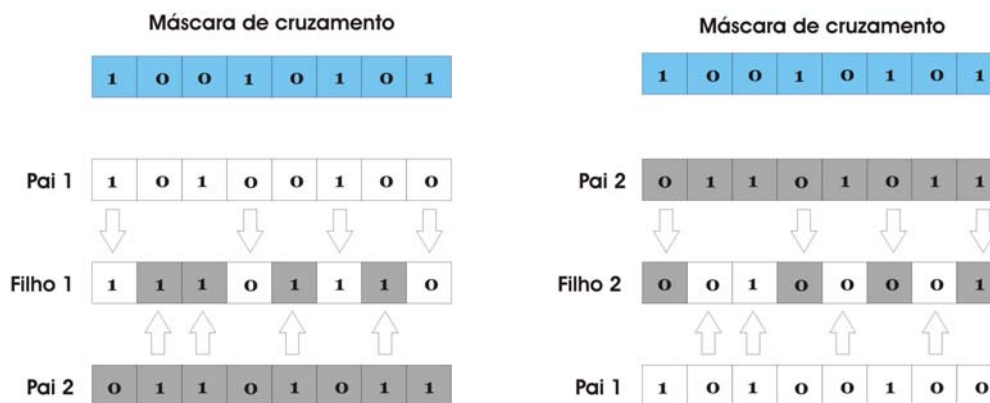


Figura 5: Esquema gráfico do cruzamento uniforme.

### 2.2.3 Mutação

A mutação é vista como o operador responsável pela introdução e manutenção da diversidade genética na população. Ela trabalha alterando arbitrariamente, logo após o cruzamento, um ou mais componentes de uma estrutura escolhida entre a descendência, fornecendo dessa forma meios para a introdução de novos elementos na população. O operador de mutação é aplicado aos indivíduos com uma probabilidade dada por uma taxa de mutação  $P_m$ . A figura 6 ilustra o processo de mutação em um indivíduo [Holland, 1975].

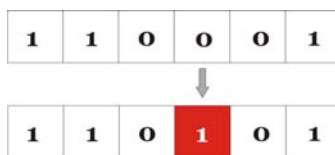


Figura 6: Esquema gráfico de ocorrência de Mutação.

A ocorrência de mutação em determinado gene é dada pela taxa de mutação, que usualmente, possui um valor pequeno, sendo alguns dos valores mais comumente usados  $P_m = 0,001$  e  $P_m = 0,1$  [Bäck, 1996]. Thomas Bäck, em suas últimas pesquisas constatou que a performance do AG tende a decair em populações de tamanho relativamente grande ( $n > 200$ ) usando grande probabilidade de mutação ( $P_m > 0,05$ ) e em populações de pequeno tamanho ( $n < 200$ ) combinadas com pequena probabilidade de mutação ( $P_m < 0,02$ ) [Bäck, 1996].

### 2.3 Parâmetros Genéticos

É importante analisar de que maneira alguns parâmetros influenciam o comportamento dos algoritmos genéticos para que se possa estabelecê-los conforme as necessidades do problema e dos recursos disponíveis. A seguir, são listados alguns parâmetros genéticos utilizados frequentemente:

- **Tamanho da População:** O tamanho da população determina o número de cromossomos na população, afetando diretamente o desempenho global e a eficiência dos AG's. Com uma população pequena, o desempenho pode cair, pois deste modo a população fornece uma pequena cobertura do espaço de busca do problema. Uma grande população geralmente fornece uma cobertura representativa do domínio do problema, além de prevenir convergências prematuras (tendência da população a evoluir para uma solução não ótima devido a existência de um indivíduo com aptidão muito superior às demais aptidões). No entanto, para se trabalhar com grandes populações, são necessários maiores recursos computacionais ou que o algoritmo trabalhe por um período de tempo muito maior.
- **Taxa de Cruzamento:** Quanto maior for esta taxa, mais rapidamente novas estruturas serão introduzidas na população. Mas se esta for muito alta, a maior parte da população será substituída, e pode ocorrer perda de estruturas de alta aptidão. Com um valor baixo, o algoritmo pode tornar-se muito lento.
- **Tipo de Cruzamento:** O tipo de cruzamento a ser utilizado determina a forma como se procederá a troca de segmentos de informação entre os "casais" de cromossomos selecionados para cruzamento. O ideal seria testar diversos tipos de cruzamento em conjunto com as outras configurações do AG para verificar qual apresenta um melhor resultado.
- **Taxa de Mutação:** A taxa de mutação determina a probabilidade em que uma mutação ocorrerá. Mutação é utilizada para introduzir novas informações na população e também para prevenir que a população se sature com cromossomos semelhantes (convergência prematura). Uma baixa taxa de mutação previne que uma dada posição fique estagnada em um valor. Com uma taxa muito alta a busca se torna essencialmente aleatória além de aumentar muito a possibilidade de que uma boa solução seja destruída. A melhor taxa de mutação é dependente da aplicação, mas, para a maioria dos casos está entre 0,001 e 0,1 [Bäck, 1996].

### 3 Estudo de caso: AMALTHAEA

O aumento da rede mundial de computadores tem como resultado mais visível o aumento da quantidade de informação disponível on-line. Esta informação é distribuída entre fontes heterogêneas e muitas vezes não estruturada. Diante desta situação, novas técnicas, como os agentes de informação, são criadas para ajudar a lidar com este excesso de informação. Para isso, os agentes devem capturar os interesses e hábitos dos usuários, por exemplo, usando técnicas de aprendizagem de máquina, tendo o cuidado de se adaptarem aos novos interesses assim como explorar novos domínios que podem ser de interesse do usuário.

O domínio em que estamos nos focando é o acesso à informação personalizada e a descoberta de novas informações. Nesse contexto é apresentado o AMALTHAEA [Moukas, 1997], um ambiente multiagente para filtragem personalizada, recuperação e monitoramento de fontes de informação, voltado para a rede mundial de computadores, a Internet. Sua principal finalidade é ajudar os usuários a encontrar informações relevantes dentre a diversidade de informações disponíveis na grande rede.

O AMALTHAEA considera três domínios em paralelo [Moukas, 1997]:

- *World Wide Web* (WWW) e descoberta de dados. AMALTHAEA não procura na WWW em si, mas ao invés disso lança agentes múltiplos que utilizam mecanismos de buscas e executa uma "meta-busca" para descobrir informação de interesse do usuário. O sistema então, analisa os documentos recuperados e usa uma técnica baseada no modelo do espaço vetorial [Salton, 1983] para selecionar aqueles mais próximos das preferências do usuário.
- Fluxo contínuo de informação e filtragem de informação. Os agentes analisam os documentos e selecionam somente os mais apropriados. O AMALTHAEA filtra os documentos que chegam continuamente, como, por exemplo, as notícias de uma determinada agência.
- Monitoramento de mudanças frequentes em fontes de informações. Às vezes o usuário quer monitorar certas URLs (*Universal Resource Location*) que são atualizadas com certa frequência. Por exemplo, ele quer saber, dentro da Fórmula 1, resultados das corridas que são atualizados a cada dois domingos, ou o usuário quer

localizar novos artigos em um diário on-line, ou novos CDs de um artista. Tais URLs são monitoradas em intervalos regulares para mudanças. Se tais mudanças ocorrerem é necessário que o usuário seja notificado.

### 3.1 A implementação básica do AMALTHAEA

A implementação do AMALTHAEA foi baseada na criação de um ecossistema artificial de agentes evolutivos que cooperam e competem em um ambiente de recursos limitados [Moukas, 1997]. Existem duas espécies gerais de agentes: os que fazem filtragem de informação, chamados agentes IF (*Information Filtering*) e os que buscam informação, chamados agentes ID (*Information Discovery*). Os agentes IF são responsáveis pela personalização do sistema e por manter o perfil do usuário. Os agentes ID são responsáveis pelo manuseio das fontes de informação, localizando e buscando a informação atual que o usuário está interessado. Quando treinados, os agentes ID tornam-se melhores, sendo capazes de especificar quais *sites* de busca são os mais apropriados para determinado assunto.

A evolução é feita através da busca constante pela melhor solução possível existente para o problema (em nosso caso, a melhor informação possível para os interesses do usuário). As soluções devem se adaptar a novas situações rapidamente (por exemplo, seguindo um novo interesse de usuário). Ao mesmo tempo, o sistema continua explorando o espaço de busca para encontrar novas soluções, usando operadores genéticos como mutação e cruzamento para atualizar e especializar a população de agentes.

Para entender o comportamento global do sistema durante a filtragem da informação, podemos considerar cada agente IF como um filtro muito especializado que só é aplicado em um setor estreito do domínio. Quando um usuário muda seus interesses, os filtros associados aos antigos interesses são destruídos e outros novos são criados, apontando para os novos interesses através de evolução e seleção natural. Uma questão que deve ser discutida em tais sistemas multiagentes é como encontrar maneiras de permitir que o sistema alcance um equilíbrio enquanto adapta-se continuamente aos interesses de novos usuários. Para avaliar o desempenho de cada agente estabeleceu-se a noção de um ecossistema que opera na base de um simples modelo econômico: os agentes que são úteis ao usuário adquirem créditos positivos, enquanto os agentes de baixa performance adquirem créditos negativos. Os agentes com melhores avaliações tendem ficar no sistema enquanto os piores serão destruídos [Clearwater, 1996].

Uma das grandes funcionalidades do AMALTHAEA é a capacidade de indicação de *sites* que sejam de interesse do usuário e isto é possível graças a sua capacidade de modelar o perfil dos usuários. O AMALTHAEA capta os interesses do usuário de quatro modos diferentes:

- Através do próprio usuário, que submete o *bookmark* contendo seus *sites* favoritos.
- O AMALTHAEA checa o histórico de páginas visitadas do usuário, arquivadas pelo *browser*. O sistema analisa esta informação e tenta deduzir certos padrões para decidir se monitora algum dos *sites*;
- Selecionando-se pacotes de agentes pré-treinados (cada pacote focaliza um tópico em particular. Por exemplo “futebol europeu”, “a Grécia”, etc.) para acelerar a aprendizagem do AMALTHAEA;
- Através de interesses específicos para treino do sistema baseado em alguns documentos importantes.

Baseado nestas informações, o sistema usa mecanismos de busca para encontrar documentos similares. AMALTHAEA recomenda esses documentos ao usuário e se aperfeiçoa através de duas formas de avaliação:

1. Direta, onde o usuário taxa um documento específico;
2. Indireta, onde o sistema observa quanto tempo um usuário permanece em uma página.

Usuários podem monitorar a operação do AMALTHAEA e ajustar seu comportamento adequadamente.

### 3.2 Arquitetura do AMALTHAEA

AMALTHAEA designa para cada usuário seus próprios agentes IF e ID e gera um ecossistema fechado. Várias pessoas podem usar o sistema, mas todos seus arquivos estão separados e nenhuma interação entre diferentes usuários acontece no sistema. Todos os componentes do sistema que serão discutidos operam em benefício de um único usuário. Para tratar vários usuários, o sistema usa várias instâncias daqueles componentes [Moukas, 1997].

O AMALTHAEA é composto por cinco módulos básicos (figura 7):

1. Uma interface com o usuário, que é feita através de um *browser* em Java e que efetua a comunicação entre o usuário e a aplicação. A interface apresenta as informações recuperadas em forma de sumário e permite que as mesmas sejam avaliadas (*feedback* do usuário);

2. Os agentes IF que tentam selecionar documentos que preencham as preferências do usuário e os agentes ID que tentam encontrar na *Web* os *sites* com informações relevantes para o usuário;
3. Mecanismos que permitem recuperar documentos da *Web*;
4. Mecanismos que processam o texto contido nas páginas, realizando a vetorização de documentos, extração de palavras irrelevantes e remoção de prefixos e sufixos;
5. Uma base de dados onde constam todas as URLs de todas os documentos considerados relevantes para o usuário.

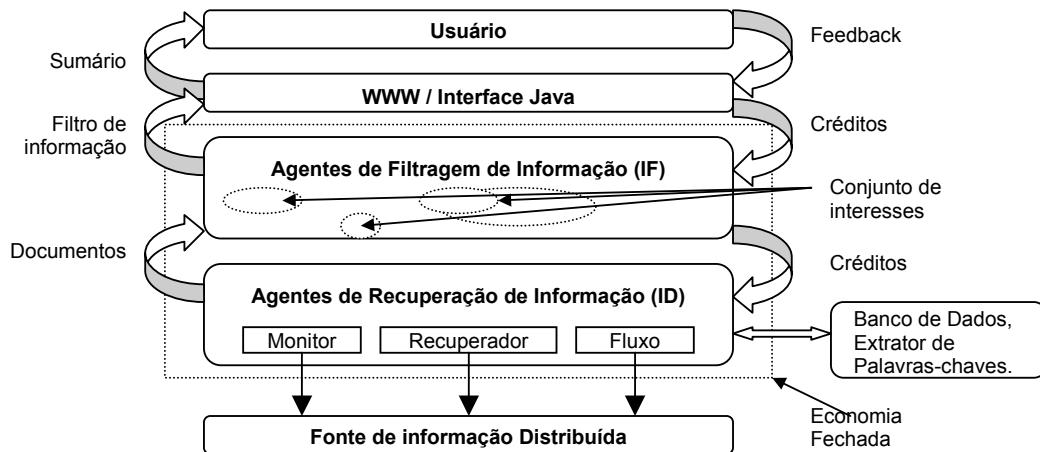


Figura 7: Arquitetura básica da AMALTHAEA

### 3.3 Representação Interna

Toda as fontes de informação (documentos de fontes como: páginas da WWW, ftp *sites*, serviços de informação baseadas na WWW, grupo de notícias, etc.) do sistema podem ser acessadas através da WWW (http, ftp, notícias). O mecanismo inicial para recuperação de documentos foi baseado na biblioteca *libwww* da WWW [Frystyk, 1994]. No topo deste mecanismo é construída uma biblioteca para normalizar URLs antes de serem armazenadas em banco de dados de “URLs já recuperadas”.

Todas as URLs recuperadas são guardadas em um banco de dados em forma de vetores ponderados de palavras-chaves. Eles são acompanhados por um *checksum* de HTML que habilita os agentes ID verificar se a URL específica sofreu mudanças desde a última visita, no caso de um *site* monitorado. Uma lista separada de *links* contidos em cada URL é gerada (até uma profundidade definida pelo usuário) para auxiliar o monitoramento de certos *sites* que agem como “pontos de partida” na exploração da Internet (por exemplo, uma lista de todos os sites relacionados por agente). Finalmente, o banco de dados é utilizado para guardar trilha das URLs já incluídas em sumários prévios para prevenir apresentação de informação duplicada ao usuário.

A representação interna dos documentos no AMALTHAEA é baseada em uma técnica de recuperação de informação padronizada chamada representação de vetor ponderado [Salton, 1983]. A representação básica dos agentes IF e dos analisadores de arquivos HTML está no vetor ponderado de palavras-chaves. Quando os arquivos HTML são processados, uma versão limpa do texto é gerada. O texto é decomposto em palavras-chaves, que são ponderadas e utilizadas para compor o vetor de palavras-chaves. A análise de similaridade entre documentos e a seleção de documentos realizada pelos agentes IF são baseados em operações sobre o vetor ponderado de palavras-chaves. Depois que o código fonte HTML de uma dada URL é recuperado, um analisador gramatical remove as *tags*, identifica o título do documento e os *links* contidos na página HTML e salva aqueles dados para serem processados depois.

O texto original está consequentemente processado para que as palavras mais importantes sejam identificadas e classificadas. A seguir apresenta-se um exemplo da evolução de uma sentença através das diferentes fases de geração do vetor:

- Texto Original: *Agents are running objects with an attitude.*

- Palavras como “the”, “it”, “he”, “will”, comuns na língua inglesa, são removidas do texto. Depois da remoção a sentença torna-se: *Agents running objects attitude*.
- As palavras restantes são derivadas, seus sufixos são removidos, deixando somente as raízes da palavra. *Agent run object attitud*

As palavras-chaves que sobrevivem ao processo de derivação são gravadas na forma de uma matriz  $M \times 2$ , onde  $M$  é o número de palavras-chaves. A primeira coluna da matriz armazena as palavras-chaves em ordem alfabética e a segunda coluna armazena sua frequência no documento. As palavras-chaves do título da página submetida e a URL da página são associadas a um peso igual a  $0,05 \cdot M^2$ , e às URLs encontradas no documento HTML são dadas um peso igual a metade de suas frequências no documento [Moukas, 1997].

Finalmente, cada palavra-chave é ponderada segundo a medida “*tfidf*”. A sigla “*tfidf*” é um acrônimo para “*Term Frequency times Inverse Document Frequency*” e é um mecanismo padronizado de ponderação de informação [Salton, 1983]:

$$W_k = H_c \cdot T_f \cdot idf_k,$$

Onde  $T_f$  é a frequência de ocorrência da palavra-chave no documento atual (frequência de termo), o  $H_c$  é uma constante cujo valor está associado a posição da palavra-chave no documento (corpo do texto, título, etc), e  $idf_k$  é frequência da palavras-chaves na coleção inteira de documentos. O termo  $idf_k$  é formalmente definido como:

$$idf_k = \log\left(\frac{N}{df_k}\right),$$

Onde  $N$  é o número total de documentos que tem um estado de “já recuperado” pelo sistema e  $df_k$  é a frequência da palavra em todos os documentos analisados até ao momento.

Neste caso, a coleção de documentos é o conjunto de todos os vetores ponderados de palavras-chaves, que são a representação interna dos documentos recuperados. A constante do cabeçalho é igual a 1,0 se a palavra-chave foi encontrada no texto do corpo do documento. Por outro lado, se a palavra-chave estava contida no título, seu peso é multiplicado por uma constante maior que 1,0. Deste modo palavras-chaves contidas no título da página terão um peso maior que aquelas verificadas no corpo do documento.

Usando o método acima, todos os documentos da WWW são representados em um espaço multidimensional.

### 3.4 Codificação e Operadores do AMALTHAEA

A evolução dos agentes é controlada por dois elementos: a aptidão individual e a aptidão global do sistema. Somente alguns elementos (os mais aptos) da população inteira têm permissão de se proliferar. A qualificação de um agente é baseada somente em sua aptidão. O número dos agentes aos que será dada permissão de produzir descendentes está linearmente relacionado ao número de agentes que serão excluídos devido à baixa performance (baixa aptidão). Esses números não são constantes e estão relacionados à aptidão global do sistema. Se a aptidão global está diminuindo então a taxa de cruzamento é aumentada em busca de uma adaptação mais rápida aos interesses de novos usuários. Se a aptidão global está aumentando, a taxa de cruzamento é mantida em um patamar baixo para permitir que o sistema lentamente explore o espaço de busca de melhores soluções [Moukas, 1997].

Novos agentes são criados através de cópia (ou clonagem), cruzamento ou mutação. Todos os operadores são aplicados na parte dos agentes que evolui, o genótipo, que contem a codificação de uma determinada solução de um problema. A outra parte dos agentes, o fenótipo, contem informações que não evoluem, geralmente instruções como: identificador do agente, usuário, etc. O operador de cópia toma os melhores agentes e cria mais agentes como eles. Através do operador cruzamento, dois agentes, os pais, retornam dois novos agentes, os filhos, que herdaram uma parte do vetor de palavras-chaves dos pais. Este operador aleatoriamente seleciona dois pontos no vetor de palavra-chave e troca todos campos dos dois pais entre esses pontos, criando dois novos agentes.

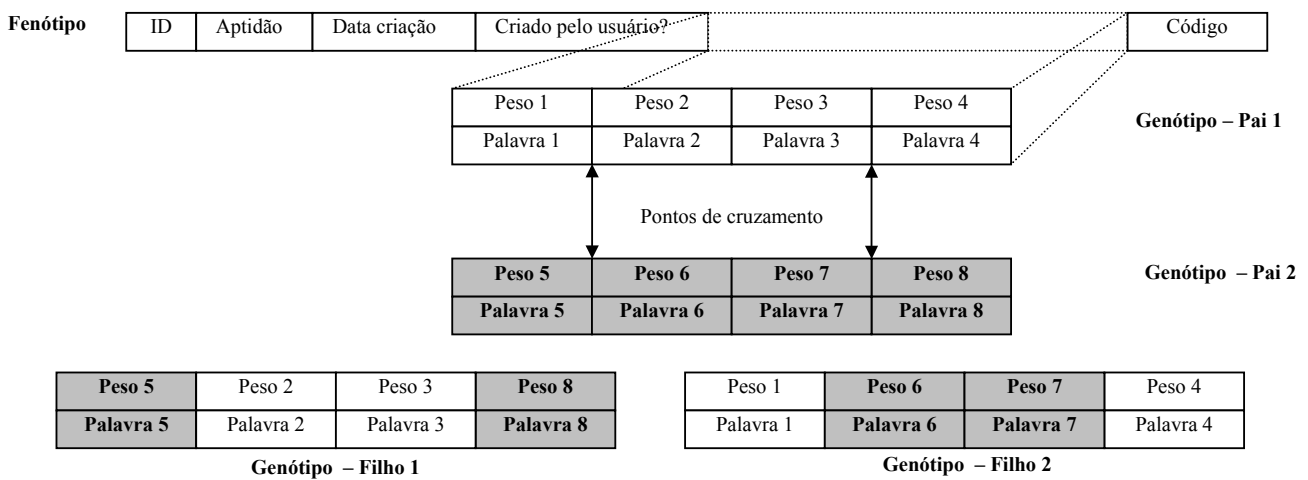
Considerando um genótipo  $G$ , os pontos de cruzamento  $\rho_1$  e  $\rho_2$  põem ser encontrados da seguinte forma:

$$\begin{cases} \rho_1 = \text{rand}(1, \text{sizeof}(G) - 2) \\ \rho_2 = \text{rand}(\rho_1, \text{sizeof}(G) - 1) \end{cases}$$

O novo genótipo herda uma parte dos vetores de palavras-chaves de seus pais. Uma variável  $i$  representa a peso da palavra-chave.

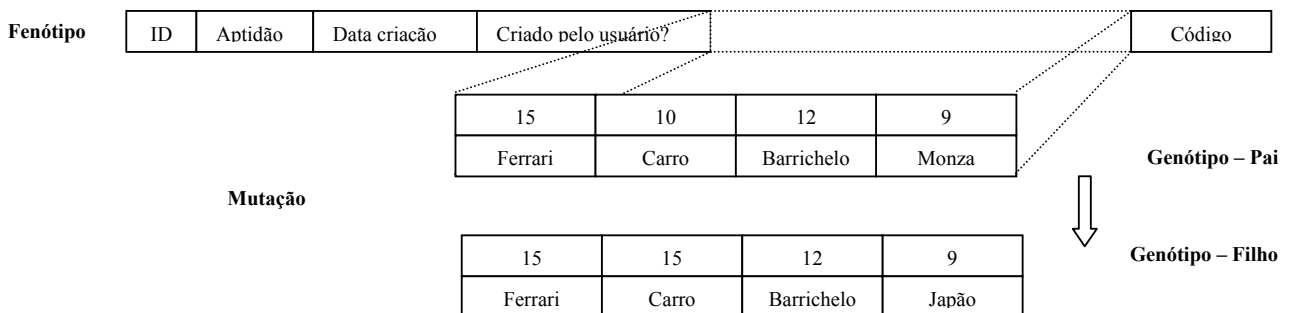
$$G_3 = \begin{cases} G_{1i}, & 0 < i < \rho_1 \text{ e } \rho_2 < i \leq \text{sizeof}(G) - 1 \\ G_{2i}, & \rho_1 \leq i \leq \rho_2 \end{cases}$$

$$G_4 = \begin{cases} G_{1i}, & \rho_1 \leq i \leq \rho_2 \\ G_{2i}, & 0 < i < \rho_1 \text{ e } \rho_2 < i \leq \text{sizeof}(G) - 1 \end{cases}$$



**Figura 8:** Esquema de um cruzamento entre Agentes IF.

A mutação é outro operador que pode ser usado tanto sozinho quanto dentro dos operadores de cruzamento e cópia. O operador de mutação toma o genótipo de um agente como argumento e cria um novo agente que é uma versão modificada aleatoriamente de seu pai. Os pesos das palavras-chaves são modificados aleatoriamente enquanto a nova palavra-chave “mutada” é aleatoriamente selecionada de um agente que pertence a outro cluster ou a uma recuperação recente. A seguir é mostrado como ocorre a mutação (figura 9).



**Figura 9:** Esquema de uma mutação em um Agente IF.

### 3.5 O Ecossistema

As interações entre os agentes IF e agentes ID controlam o comportamento global do sistema. A evolução é feita através da adaptação dos agentes aos interesses do usuário e está baseada em uma abordagem chamada “Controle Baseado no Mercado” [Clearwater, 1996]. Controle Baseado no Mercado é um paradigma usado para controlar sistemas complexos, já que tais sistemas podem ser difíceis de controlar, manter ou expandir. Esta forma de controle visualiza o sistema como uma miniatura de economia.

Os agentes que compõem o ecossistema operam segundo uma estratégia de penalidade/recompensa, apoiada pela noção de “crédito” que é designado indiretamente pelo usuário baseado no desempenho do sistema. O usuário dá um *feedback* ao verificar um documento no sumário. O sistema relaciona este *feedback* ao agente IF que propôs o documento e o agente ID que o recuperou e atribui um crédito. Este crédito serve como uma função de aptidão em ambas as populações. Os agentes mais aptos têm mais chances de conseguir sobreviver e produzir descendentes.

Se o *feedback* do usuário é positivo, o agente IF que propôs o artigo, recebe uma quantia de créditos diretamente relacionada com o nível de coerência de sua proposta. Se, por outro lado, o *feedback* do usuário é negativo o agente recebe créditos negativos, o que é ruim para sua aptidão. Os agentes IF “pagam” uma porcentagem fixa da quantia dos créditos que recebem aos agentes ID.

É evidente que nem todos os agentes IF podem apresentar algo a cada sumário. Os documentos propostos pelos agentes são ordenados por nível de relevância e os melhores são selecionados e apresentados ao usuário. Embora os agentes que apresentam os documentos não sejam sempre os mesmos, eles normalmente representam os melhores 40% da população. Para acelerar a destruição de agentes pouco adaptáveis e a criação de novos agentes, é introduzida uma função de decadência linear. Para que os agentes habitem o ecossistema eles têm que pagar algo. Se os créditos que eles ganham excedem este “aluguel” então eles vivem. Caso contrário eles são removidos e são criados novos agentes. Além disso, se dois agentes propõem o mesmo documento então eles recebem uma penalidade para desencorajar este tipo de ocorrência e aumentar a diversidade da população. Os agentes IF recebem avaliações diretamente do usuário.

A evolução dos agentes IF tem uma particularidade: a população, como um todo, não é evolui junta; ao invés disso, agentes IF competem entre si dentro de um determinado agrupamento. Assim todo os agentes IF que pertencem ao grupo “Grécia”, por exemplo, evoluem juntos, os agentes de “Ciência da Computação” evoluem juntos e assim por diante.

A partir de uma consulta, o fluxo de informação no AMALTHAEA é o seguinte:

- Os agentes IF enviam pedidos de documentos para os agentes ID na forma de um vetor de palavras-chaves.
- Os agentes ID selecionam o pedido do agente IF e baseado em seus fenótipos tentam recuperar documentos pertinentes utilizando os índices de mecanismos de busca na WWW a que são associados.
- Cada agente ID apresenta a seu agente IF o conjunto de documentos recuperados e o agente IF seleciona os melhores, segundo seu vetor de palavras-chaves.
- Vários agentes IF incluem os seus documentos selecionados no sumário do usuário baseado no grau de aptidão de seus documentos.
- O usuário classifica os documentos apresentados no sumário e dá um *feedback* ao sistema.

## 4 Conclusão

Devido às suas características de robustez, flexibilidade e relativa facilidade de implementação, os Algoritmos Genéticos irão ganhar uma maior atenção com o decorrer do tempo, principalmente, pela rápida evolução dos computadores que tornarão as aplicações da técnica cada vez mais viáveis e engenhosas.

Embora este trabalho tenha dado atenção a aplicação dos algoritmos genéticos no acesso a informação personalizada, inúmeras outras aplicações da técnica são possíveis e já foram empregadas, onde podemos citar como exemplo na construção automática de programas para a realização de tarefas específicas e na criação de imagens, texturas, músicas e estruturas complexas.

A realização deste trabalho permitiu traçar o panorama existente em relação a quantidade de informação com que lidamos diariamente e a dificuldade que encontramos para nos atualizar diante da constante evolução. A criação de ferramentas para ajudar no tratamento e recuperação de informações, assim como técnicas que possam modelar interesses dos usuários de forma proativa tem se tornado objeto de pesquisa

Os resultados deste trabalho estão sendo aplicados no contexto do projeto MaAE [Girardi, 2001] [Girardi, 2002]. Agentes deliberativos reutilizáveis para a modelagem de usuários e acesso à informação personalizada baseados em algoritmos genéticos estão sendo desenvolvidos [Filho, 2003].

Por sua vez, essa experiência de desenvolvimento está sendo utilizada para a extração e descrição de um conjunto de padrões de software para o projeto de sistemas adaptativos baseados na modelagem de usuários [Girardi 2003] [Oliveira, 2003].

## 5 Agradecimentos

Este trabalho tem o apoio do CNPq.

## 6 Bibliografia

- [Bäck, 1996] Bäck, T. "Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms". New York:Oxford University Press, 1996.
- [Clearwater, 1996] Clearwater, S. "Market Based Control: a paradigm for distributed resource allocation". A comparative-developmental approach to understanding imitation. AAAI Press, 1996.
- [Darwin, 1859] Darwin, C. "Sobre a Origem das Espécies por Meio da Seleção Natural". 1859.
- [Filho, 2003] Filho, J. A. R. S. "Uma Análise e Exemplo de Aplicação das Técnicas de Aprendizagem de Máquina Baseados em Algoritmos Genéticos". Relatório do Estudo Orientado. Curso de Pós-graduação em Engenharia Elétrica. UFMA, 2003.
- [Fisher, 1960] Fisher, R.A. "The Genetic Theory of Natural Selection". 1960.
- [Frystyk, 1994] Frystyk, H. & Lie, H. "Towards a uniform library of common code". Em: Proceedings of the Second WWW Conference, 1994.
- [Girardi, 2001] Girardi, M. D. R. "Agent-Based Application Engineering". Em: 3<sup>rd</sup> International Conference On Enterprise Information Systems (ICEIS 2001), Setúbal, 2001
- [Girardi, 2002] Girardi, M. D. R. "Reuse in Agent-based Application Development". Em: 1<sup>st</sup> International Workshop On Software Engineering For Large-Scale Multi-Agent Systems (SELMAS'2002), May 2002
- [Girardi, 2003] Girardi, R., Oliveira, I. e Bezerra, G. Towards a System of Patterns for the Design of Agent-based Systems, Proceedings of The Second Nordic Conference on Pattern Languages of Programs" (VikingPLOP 2003). Bergen, Norway. 19 de setembro de 2003.
- [Goldberg, 1989] Goldberg, D. E. "Genetic Algorithms in Search, Optimization and Machine Learning". Massachusetts: Addison-Wesley Co, 1989.
- [Holland, 1975] Holland J. H. "Adaptation in Natural and Artificial Systems". Ann Arbor:University of Michigan Press, 1975.
- [Mitchell, 1996] Mitchell, M. An Introduction to Genetic Algorithms. Massachusetts: MIT Press, 1996.
- [Moukas, 1997] Moukas, A. G. "Amalthaea: Information Filtering and Discovery Using A Multiagent Evolving System". Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning, in partial fulfillment of the requirements for the degree of Master of Science in Media Technology at the Massachusetts Institute of Technology June 1997.
- [Oliveira, 2003] Ismênia de Oliveira e Rosario Girardi. Padrões Arquiteturais e de Projeto para a Modelagem de Usuários baseada em Agentes, Anais da Terceira Conferência Latinoamericana em Linguagens de Padrões para Programação (SugarLoafPLOP 2003). Porto de Galinhas, Pernambuco, Brasil. 12 a 15 de agosto de 2003. A ser publicado.
- [Palazzo, 1997] Palazzo, L. A. M. "Algoritmos para Computação Evolutiva". - Relatório Técnico - Pelotas: Grupo de Pesquisa em Inteligência Artificial - Universidade Católica de Pelotas.

- [Salton, 1983] Salton, G. & McGill, M. J. "Introduction to Modern Information Retrieval". McGraw-Hill, New York, 1983.
- [Schneider, 1998] Schneider, A. M. "Algoritmo Adaptativo Genético para Acompanhamento da Trajetória de Alvos Móveis" Dissertação de Mestrado - Porto Alegre: Instituto de Informática - Universidade Federal do Rio Grande do Sul, 1998.
- [Sobrinho, 2003] Sobrinho, A. C. "Uma análise dos algoritmos genéticos e suas aplicações em sistemas de acesso à informação". Monografia de conclusão de curso, CGCC, Universidade Federal do Maranhão, 2003.
- [Shapiro, 1999] Shapiro, J. "Genetic Algorithms in Machine Learning", taught at the Advanced Summer School on Machine Learning and Applications, 1999.