Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Engenharia de Computação e Automação Industrial

"COMPUTAÇÃO EVOLUTIVA EMPREGADA NA RECONSTRUÇÃO DE ÁRVORES FILOGENÉTICAS"

OCLAIR GALLACINI PRADO

ORIENTADOR: PROF. DR. FERNANDO JOSÉ VON ZUBEN
(DCA – FEEC / Unicamp)

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação (FEEC / Unicamp) como parte dos requisitos exigidos para obtenção do título de Mestre em Engenharia Elétrica.

Banca Examinadora:

Prof. Dr. Ivan Luiz Marques Ricarte (DCA - FEEC / Unicamp)
Prof. Dr. Marco Aurélio Amaral Henriques (DCA - FEEC / Unicamp)
Prof. Dr. Sérgio Furtado dos Reis (Departamento de Parasitologia / IB / Unicamp)

CAMPINAS
ESTADO DE SÃO PAULO – BRASIL
DEZEMBRO de 2001

Resumo

Esta dissertação apresenta um estudo a respeito de um dos problemas da bioinformática: a reconstrução da árvore filogenética que melhor represente um conjunto de seqüências de bases. O espaço de solução deste problema apresenta custo fatorial. Este é um problema de otimização combinatória. A Computação Evolutiva, um dos paradigmas que compõem a inteligência computacional, se mostrou uma boa ferramenta para tratar este problema devido ao seu elevado poder de exploração e explotação do espaço de soluções.

Para possibilitar a implementação de alguns dos métodos de reconstrução de árvores filogenéticas disponíveis, foi desenvolvida uma ferramenta computacional denominada "Projeto Árvores Filogenéticas", que apresenta as seguintes características:

- possui interface gráfica amigável que foi projetada para facilitar sua execução por usuários de computador, mesmo que estes não dominem completamente as técnicas de computação;
- permite ajuste dos principais parâmetros relativos à filogenia, tais como seleção do método de busca a ser utilizado (Matriz de Distâncias ou Máxima Verossimilhança) e modelo de substituição de bases (Jukes-Cantor, 1969; Felsenstein, 1981 ou Kimura, 1980);
- permite ajuste dos principais parâmetros relativos ao algoritmo genético utilizado, tais como tamanho da população e porcentagem de mutações;
- trabalha internamente com grafos para representar as árvores e de modo a preservar informações relativas à hierarquia (ancestrais e descendentes);
- usa novos operadores especiais de mutação, desenvolvidos para trabalhar com as matrizes de adjacências dos grafos.

É interessante ressaltar que o Projeto Árvores Filogenéticas utiliza um procedimento computacional inspirado na evolução natural para buscar uma boa explicação para os efeitos da própria evolução natural, na forma de uma árvore filogenética. Este projeto se encontra disponível no seguinte endereço: ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/oclair/.

Abstract

This dissertation presents a study about one of the bioinformatic problems: the phylogenetic tree reconstruction. The solution space for this problem is calculated using a factorial formula. This is a combinatorial optimization problem. Evolutionary Computation, one of the component paradigms of computational intelligence, was proved to be a good tool to tackle this problem, due to its high potential to explore and exploit solution spaces.

In order to allow the implementation of some methods of phylogenetic tree reconstruction, a computational tool named “Phylogenetic Tree Project”, was developed and has the following features:

- has a friendly graphical user interface designed mainly to assist the user during the execution, even the ones that do not have enough background in computer sciences;
- allows the adjustment of phylogenetic special purpose parameters, such as the method of search (Distance Matrix or Maximum Likelihood) and base substitution model (Jukes-Cantor, 1969; Felsenstein, 1981 or Kimura, 1980);
- allows the adjustment of genetic algorithm special purpose parameters, such as the amount of individuals in the population and the mutation percentage;
- works internally with graphs to represent the phylogenetic trees, and such that graph structure is able to preserve the hierarchical information (ancestors and descendants);
- new special mutation operators were developed to operate over the graphs adjacency matrices.

It is interesting to reinforce that the software called “Phylogenetic Tree Project” uses a method inspired by the natural evolution to search for a good explanation for the effects of natural evolution itself based on a phylogenetic tree.

This toolbox is available at ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/oclair/.

Dedico aos meus pais,
Adauto & Amélia.

Agradecimentos

Em primeiro lugar agradeço à minha esposa Isabel pelo apoio, carinho e dedicação nestes últimos anos, sem o qual eu não teria sido capaz de finalizar meus estudos. Agradeço também aos meus filhos Erick e Cynthia por terem sido sempre crianças maravilhosas e por terem respeitado a finalidade do tempo que não passei com eles.

Serei sempre grato ao meu orientador, o Professor Fernando José Von Zuben. O Professor Von Zuben, com seus profundos conhecimentos na área de inteligência computacional, norteou de forma decisiva este trabalho. Sua dedicação, a atenção dispensada aos seus orientados e sua determinação para o trabalho constituem um exemplo que certamente influenciarão o resto de minha vida.

Agradeço aos Professores Márcio Andrade Netto e Sérgio Furtado dos Reis pelo grande incentivo e valiosas contribuições para minha formação nestes últimos anos na Pós-Graduação da Unicamp.

Sou grato aos colegas Marcelo Marques Gomes e Artemis Moroni que colaboraram nas mais diversas formas para o desenvolvimento deste trabalho, desde profícuas discussões até mesmo através de sua amizade e companheirismo.

Aos amigos Márcio Henrique Zuchini e Paulo Eduardo de Campos Gaspar agradeço o carinho e a amizade, que foram de grande importância durante o decorrer deste trabalho.

Agradeço ao Departamento de Engenharia de Computação e Automação Industrial, Faculdade de Engenharia Elétrica e de Computação, Unicamp, pela oportunidade de fazer o curso de Mestrado.

Por fim, agradeço à Fundação CPqD pelo auxílio e apoio concedidos aos meus estudos.

Índice

RESUMO ..... **III**

ABSTRACT ..... **IV**

DEDICATÓRIA ..... **V**

AGRADECIMENTOS ..... **VII**

ÍNDICE ..... **IX**

CAPÍTULO 1 INTRODUÇÃO **1**

1.1 INTELIGÊNCIA COMPUTACIONAL 2

1.2 COMPUTAÇÃO EVOLUTIVA 3

1.3 FERRAMENTA COMPUTACIONAL DESENVOLVIDA 3

1.4 ORGANIZAÇÃO DA DISSERTAÇÃO 4

CAPÍTULO 2 COMPUTAÇÃO EVOLUTIVA **7**

2.1 INTRODUÇÃO 7

2.2 IDÉIAS EVOLUCIONISTAS DE DARWIN 8

2.3 TEORIA DA EVOLUÇÃO NATURAL 9

2.4 COMPUTAÇÃO EVOLUTIVA 12

2.5 ALGORITMOS GENÉTICOS 15

2.5.1 Codificação de indivíduos: representação genética 16

2.5.2 Formação da população inicial 18

2.5.3 Função de avaliação 18

2.5.4 Operadores genéticos 19

2.5.5 Valores para os diversos parâmetros 24

2.5.6 Técnicas de seleção 25

2.6 DIVERSIDADE POPULACIONAL E DERIVA GENÉTICA 27

2.7 REPRESENTAÇÃO EM ÁRVORE E OPERADORES GENÉTICOS ASSOCIADOS 33

2.8 ALGORITMOS MEMÉTICOS 40

CAPÍTULO 3 FILOGENIA **47**

3.1 INTRODUÇÃO 47
3.2 REPRESENTAÇÃO DA FILOGENIA 47
3.3 ÁRVORE GÊNICA E ÁRVORE DE ESPÉCIES 48
3.4 ÁRVORE COM RAIZ E SEM RAIZ 49
3.5 MÉTODOS BASEADOS E NÃO BASEADOS EM MODELO 49
3.6 MÉTODOS NÃO BASEADOS EM MODELO 50
3.6.1 Matriz de Distâncias 50
3.6.1.1 Método UPGMA 50
3.6.1.2 Quadrados mínimos 53
3.6.1.3 Evolução Mínima 54
3.6.1.4 Neighbor-Joining 55
3.6.2 Máxima Parcimônia 56
3.7 MÉTODOS BASEADOS EM MODELO 60
3.7.1 Máxima Verossimilhança 60

CAPÍTULO 4 ANÁLISE FILOGENÉTICA 65

4.1 INTRODUÇÃO 65
4.2 ESTUDO DE ALGUNS TRABALHOS SOBRE FILOGENIA 66
4.2.1 Antes do uso de Algoritmos Genéticos 66
4.2.2 Primeira referência sobre Algoritmos Genéticos para o problema das árvores filogenéticas 68
4.2.3 Lewis (1998) 68
4.2.4 Reijmers et al. (1999) 69
4.2.5 Skourikhine (2000) 70
4.2.6 Goloboff & Farris (2001) 70
4.2.7 Janis & Wheeler (2001) 71
4.2.8 Moilanen (2001) 72

CAPÍTULO 5 RESULTADOS EXPERIMENTAIS DA PESQUISA 77

5.1 INTRODUÇÃO 77
5.2 ESTUDO DE ÁRVORES FILOGENÉTICAS USANDO COMPUTAÇÃO EVOLUTIVA 79

CAPÍTULO 6 CONCLUSÕES E PERSPECTIVAS FUTURAS 107

6.1 COMENTÁRIOS GERAIS SOBRE O TRABALHO 107

6.2 CONCLUSÕES 110
6.3 CONTRIBUIÇÕES 112
6.4 TEMAS SUGERIDOS COMO PERSPECTIVAS FUTURAS DA PESQUISA 113

REFERÊNCIAS BIBLIOGRÁFICAS 117

ÍNDICE DE AUTORES 125

APÊNDICE A MANUAL DO USUÁRIO DO PROJETO ÁRVORES FILOGENÉTICAS 129

APÊNDICE B CÁLCULO DO NÚMERO DE ÁRVORES, RAMOS E NÓS DE UMA ÁRVORE 167

APÊNDICE C ESTRUTURAS DE DADOS 171

Capítulo 1

Introdução

Conforme pode ser encontrado em Haeckel (1866), desde o tempo de Charles Darwin tem sido um sonho de muitos cientistas a reconstrução da história evolucionária de todos os organismos da Terra, expressando-a na forma de uma árvore filogenética.

"Árvore Filogenética" é um grafo estruturado (veja apêndice C, "Estruturas de dados", para maiores detalhes) na forma de uma árvore que explica o processo evolucionário e é construída a partir de seqüências de atributos (tais como seqüências de DNAs) obtidos de diversos organismos (Matsuda *et al.*, 1999).

Este trabalho procura ratificar que a Computação Evolutiva é uma ferramenta muito poderosa para auxiliar na busca de boas árvores filogenéticas, de acordo com algum critério de desempenho. É uma pretensão muito grande tentar encontrar a melhor árvore dentro de um espaço de solução gigantesco como este. Existem $2{,}75 * 10^{76}$ possibilidades quando são estudados os DNAs de somente 50 indivíduos que comporão as folhas desta árvore (Weir, 1996).

A reconstrução de árvores filogenéticas é um problema computacional muito difícil de ser tratado. Para apenas 14 elementos, o número de árvores candidatas chega a sete trilhões. Isto praticamente elimina a possibilidade de ser usada busca exaustiva e também muitos outros paradigmas de busca. Por esta razão, os métodos de inferência filogenéticas normalmente se baseiam em algoritmos de agrupamento ou estratégias heurísticas de busca que permitem minimizar a quantidade de tempo gasta analisando as árvores candidatas (Lewis, 1998).

Os métodos que são normalmente usados para estimar as árvores filogenéticas podem ser agrupados em quatro classes principais: Matrizes de Distâncias, Parcimônia, Invariantes e Máxima Verossimilhança (Swofford & Olsen, 1990). Os métodos de agrupamentos filogenéticos estudados neste trabalho são os da primeira e quarta classes. Esta restrição foi imposta somente pela necessidade de dimensionar adequadamente o tempo de execução da pesquisa. A exploração dos outros dois métodos fica sugerida como tema para extensão desta pesquisa.

Neste trabalho, usa-se uma estratégia de busca que reduz muito o tempo computacional necessário para encontrar uma boa árvore filogenética, com base no critério da Máxima

Verossimilhança: trata-se da Computação Evolutiva, ou mais precisamente os Algoritmos Genéticos.

Uma das vantagens de se usar os algoritmos genéticos é poder trabalhar com múltiplos pontos iniciais de busca e também poder operar com todos eles simultaneamente (Reijmers *et al.*, 1999).

Apesar de se inspirar fortemente em teorias vinculadas às ciências biológicas, os algoritmos genéticos têm contribuído muito pouco na resolução de problemas de reconstrução de árvores filogenéticas (Lewis, 1998).

Para efeitos desta pesquisa, cada árvore candidata é considerada como sendo um indivíduo em uma dada geração do processo evolutivo, o qual deixará, em média, mais descendentes do que os demais indivíduos da população se sua capacidade de resolver o problema (*fitness*) for relativamente maior que a dos outros.

Ao longo desta dissertação existe o interesse em demonstrar os principais aspectos da Computação Evolutiva, seu contexto histórico e científico, suas vantagens e desvantagens, bem como algumas áreas em que seu uso é altamente recomendável e outras em que ele é completamente ineficaz.

Seu alto poder de busca e exploração torna possível encontrar uma boa solução para o problema de reconstruir árvores filogenéticas, o qual apresenta uma explosão combinatória de possíveis candidatos, como pode ser verificado em Weir (1996). Neste cenário, onde o uso de técnicas que garantem a obtenção da solução ótima conduz à intratabilidade computacional, a Computação Evolutiva, um dos ramos de atuação da Inteligência Computacional, pode se apresentar como uma alternativa eficaz.

1.1 Inteligência Computacional

A Inteligência Computacional ou "*soft computing*", usando a terminologia de Lotfy Zadeh (Zadeh, 1965), foca os aspectos sub-simbólicos do processamento de informação para criar sistemas que exibem algum grau de adaptabilidade, tolerância a falhas e também que adotam estratégias de representação e uso do conhecimento próximas dos seres humanos, conforme Bezdek (1994).

Esta área de pesquisa compreende paradigmas computacionais que procuram desenvolver sistemas que apresentam alguma forma de inteligência similar à exibida por determinados sistemas biológicos (incluindo seres humanos). Computação Evolutiva, Redes

Neurais Artificiais e Sistemas Nebulosos são os paradigmas que atualmente compõem a Inteligência Computacional, segundo Bäck *et al.* (1997).

Métodos baseados nestes paradigmas, embora poderosos na abordagem de problemas de elevada complexidade, invariavelmente demandam uma quantidade grande de recursos computacionais, seja no desenvolvimento ou no uso das ferramentas de solução. Entretanto, nos últimos anos, temos observado um desenvolvimento estrondoso da tecnologia de microprocessadores e armazenagem de informação e, como conseqüência, temos à nossa disposição cada vez mais recursos computacionais a baixo custo. Este fato tem levado a um interesse crescente da comunidade científica nos métodos baseados em Inteligência Computacional, como pode ser verificado em Iyoda (2000).

1.2 Computação Evolutiva

Computação Evolutiva é um termo relativamente recente. Ele foi proposto em torno de 1991, conforme Bäck *et al.* (1997), e representa o esforço de aproximar os pesquisadores que estavam seguindo caminhos diferentes para simular os vários aspectos da evolução. Os algoritmos genéticos, estratégias evolutivas e programação evolutiva possuem algo fundamental em comum: todos eles usam (1) reprodução, (2) variações aleatórias, (3) competição, e (4) seleção de competidores dentro de uma população. Estes elementos formam a essência da evolução e, uma vez que estes quatro processos estejam atuando na natureza ou em um computador, a evolução é o resultado inevitável, segundo Atmar (1994).

1.3 Ferramenta computacional desenvolvida

Ao longo desta pesquisa de mestrado foi criado um pacote de software denominado **Projeto Árvores Filogenéticas**, que foi desenvolvido com o propósito de se obter os resultados experimentais para esta pesquisa e também visando suprir as necessidades de usuários que necessitem reconstruir árvores filogenéticas. Para tanto, esforços significativos foram dedicados à implementação da interface com o usuário. Esta interface gráfica amigável, associada ao fato desta ferramenta fornecer sugestões para seus principais parâmetros, a torna um bom apoio mesmo para pesquisadores que não dominem completamente as técnicas de computação.

Um ponto em comum encontrado nas ferramentas disponíveis para pesquisa em bioinformática é que elas são de difícil utilização por não possuírem facilidades como estas

apresentadas pelo “Projeto Árvores Filogenéticas”. Por esta razão esta ferramenta computacional deverá ser disponibilizada para outros pesquisadores desta área.

1.4 Organização da Dissertação

O Capítulo 2 apresenta os aspectos principais relacionados à Computação Evolutiva, dando ênfase aos algoritmos genéticos, suas aplicações e restrições.

O Capítulo 3 discute os principais aspectos relacionados às análises filogenéticas, às estruturas de dados normalmente utilizadas e a métodos de reconstrução das árvores baseados em modelo (Máxima Verossimilhança) e os não baseados em modelo (Matriz de Distâncias e Máxima Parcimônia).

O Capítulo 4 procura caracterizar a análise filogenética como um problema computacional e realiza revisão bibliográfica das técnicas já empregadas para reconstrução de árvores filogenéticas. Também apresenta vários motivos que levam a crer que os algoritmos genéticos constituem uma boa ferramenta para este tipo de problema.

No Capítulo 5 são apresentados os resultados experimentais das simulações realizadas com apoio do pacote de software desenvolvido para este fim. Este software é apresentado mais detalhadamente no Apêndice A.

O Capítulo 6 apresenta as conclusões e sugere algumas perspectivas futuras para a pesquisa.

As principais contribuições deste trabalho consistem em uma nova reflexão sobre alguns dos principais conceitos ligados à filogenia, que pertence ao campo de pesquisa da Biologia, sob uma ótica voltada para otimização de sistemas e clusterização, que pertencem ao campo da Engenharia. Alguns trabalhos usados como base para esta pesquisa se mostraram extremamente enriquecedores em relação aos conceitos ligados ao campo da Biologia mas foram de pouco valor para o campo da Engenharia e o contrário também foi sentido com bastante freqüência. Neste trabalho, procuramos estabelecer um equilíbrio entre estes dois campos, esclarecendo os conceitos pertinentes a ambos e apontando as particularidades relevantes para o entendimento do problema estudado: a construção de árvores filogenéticas usando computação evolutiva.

O Apêndice A apresenta maiores detalhes sobre as telas da interface gráfica do software e também traz um exemplo que explica passo a passo os comandos e os parâmetros necessários para o processamento de um arquivo com seqüências de DNAs, desde sua

confecção usando um editor de texto comum até a análise das telas de resultados comparativos.

As fórmulas utilizadas para os cálculos dos comprimentos dos ramos bem como os algoritmos empregados para o processamento das árvores segundo os princípios da computação evolutiva serão descritos em detalhes ao longo deste trabalho.

A Figura 1.1 apresenta uma tela do **Projeto Árvores Filogenéticas** onde é exibida uma árvore determinada pelo método da Máxima Verossimilhança e que teve os comprimentos de seus ramos calculados segundo os modelos de substituição de bases de Jukes-Cantor (1969), Felsenstein (1981) ou Kimura (1980).

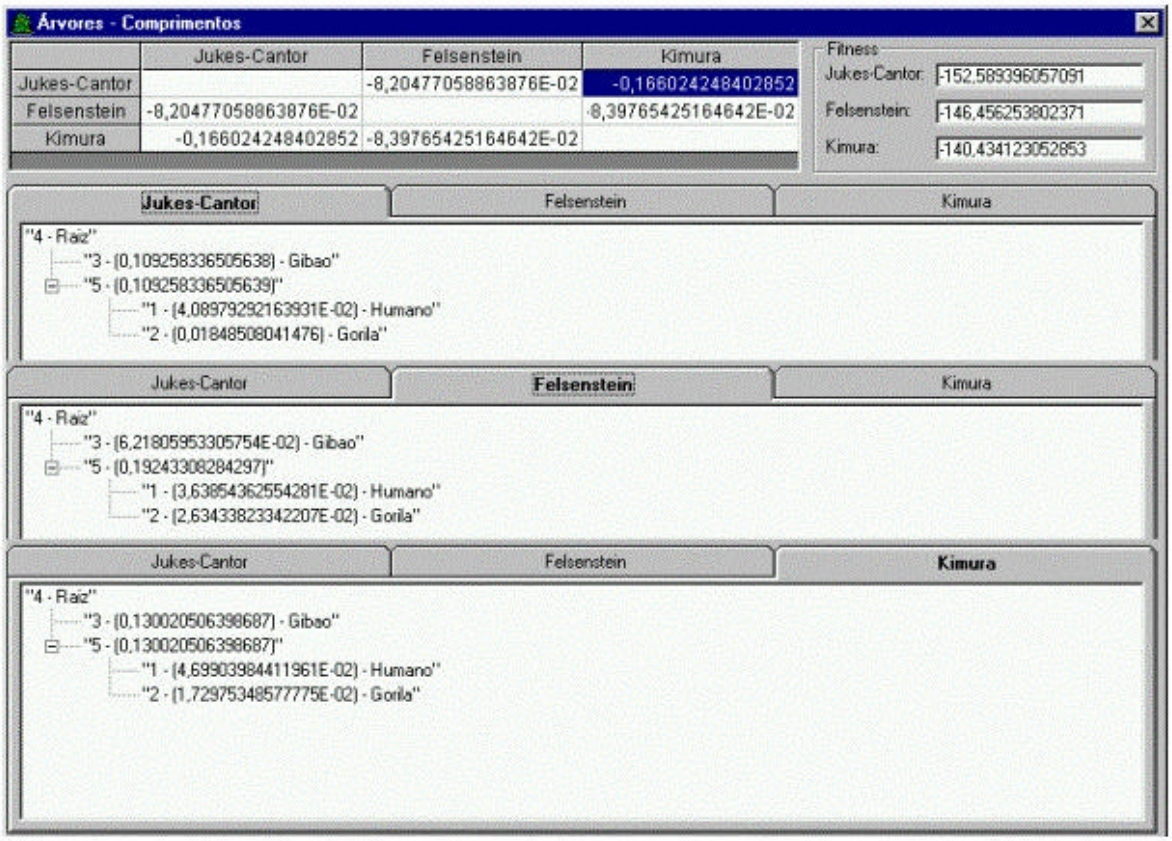


Figura 1.1 Árvores obtidas utilizando seqüência de DNA descrita por Weir (1996). Neste teste, os DNAs foram processados com o método da Máxima Verossimilhança. As três divisões da tela apresentam os resultados obtidos com o uso dos modelos de substituição de bases de Jukes-Cantor, Felsenstein e Kimura, respectivamente.

A Figura 1.2 apresenta a tela principal do **Projeto Árvores Filogenéticas** onde podem ser conferidos os parâmetros de execução do aplicativo e os resultados do processamento relativos ao tempo de execução e aos critérios de parada, tais como tempo máximo permitido para busca e quantidade máxima de gerações.

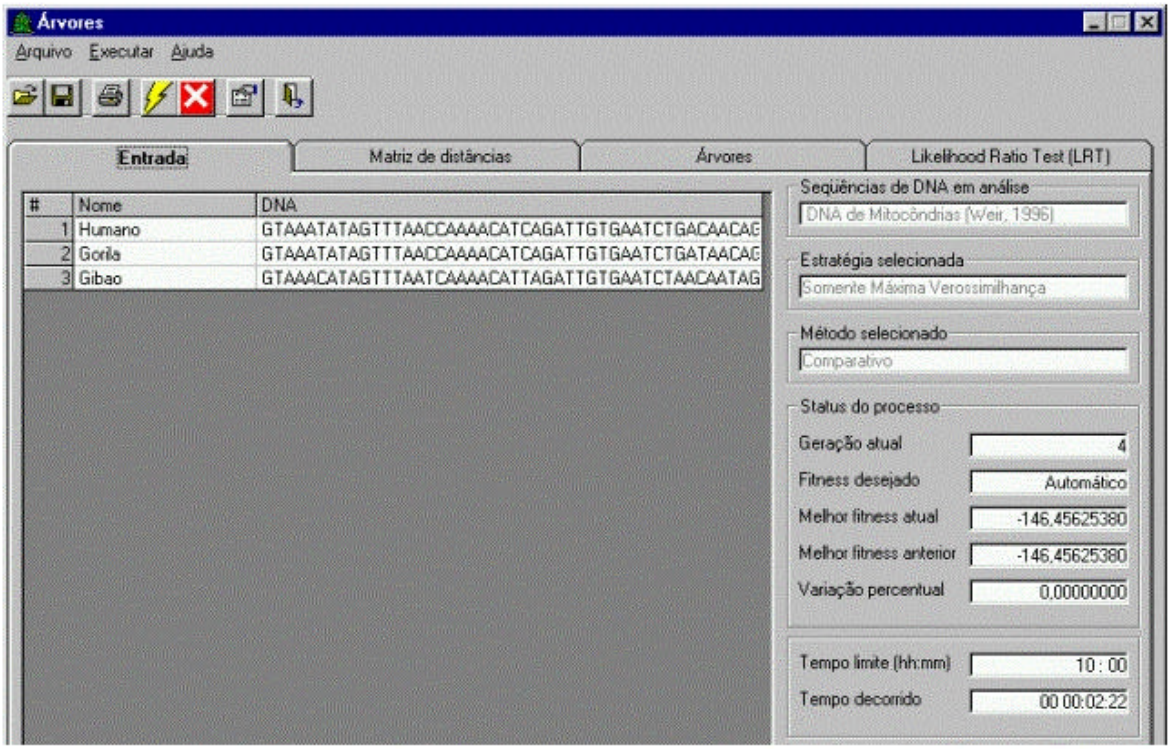


Figura 1.2 Tela principal do Projeto Árvores Filogenéticas, exibindo as seqüências de DNAs utilizadas, o tempo decorrido durante a busca e mais alguns parâmetros estabelecidos pelo usuário. "Somente Máxima Verossimilhança" significa que a busca foi realizada usando somente o método da Máxima Verossimilhança com algum modelo de substituição de bases (Jukes-Cantor,Felsenstein ou Kimura).

Ao final do Capítulo 4 são apresentados alguns resultados comparativos com outras ferramentas para este fim. Pelos motivos explicados no Capítulo 4, as comparações foram todas realizadas de maneira indireta.

Capítulo 2

Computação Evolutiva

2.1 Introdução

Segundo Bäck *et al.* (1997), o termo "Computação Evolutiva" é muito recente, tendo sido criado em 1991. Ele representa os esforços no sentido de se definir uma área comum de pesquisa que englobe os pesquisadores que vinham seguindo caminhos diferentes para simular os vários aspectos da evolução. As técnicas de "algoritmos genéticos", "estratégias evolutivas" e "programação evolutiva" possuem algo fundamental em comum: cada uma delas trata reprodução, variação aleatória, competição e seleção de indivíduos de uma população. Estes quatro elementos formam a essência da evolução (Atmar, 1994).

Se existir algum método específico que resolva um dado problema em estudo, estratégias baseadas em computação evolutiva não devem ser usadas, pois provavelmente não serão competitivas. Por não requerer informação de alta qualidade a respeito do problema, como por exemplo um modelo paramétrico que relacione entradas e saídas, elas não conseguem resolvê-lo de maneira mais eficiente nem com menor esforço computacional. Elas somente devem ser usadas para os problemas que não podem ser tratados com abordagens clássicas ou dedicadas (Bäck *et al.*, 1997).

Na Seção 2.2 são apresentadas as idéias mais importantes de Charles Darwin relativas à evolução; na Seção 2.3 são discutidas algumas idéias referentes à teoria da evolução natural; na Seção 2.4 é enfocada a computação evolutiva com descrição sucinta das principais abordagens presentes na literatura; e na Seção 2.5 são enfocados os algoritmos genéticos com mais detalhes.

2.2 Idéias Evolucionistas de Darwin

De acordo com Amabis & Martho (1997), as idéias evolucionistas de Charles Darwin podem ser resumidamente enunciadas em três conclusões, sendo apoiadas em quatro observações:

Primeira observação: as populações naturais de todas as espécies tendem a crescer rapidamente, pois o potencial reprodutivo dos seres vivos é muito grande (o potencial reprodutivo dos seres vivos deve ser maior que a taxa de mortalidade). Isso pode ser verificado, por exemplo, quando se criam determinadas espécies em cativeiro: ao garantir condições ambientais favoráveis ao desenvolvimento, sempre se observa a elevada capacidade reprodutiva inerente às populações biológicas.

Segunda observação: o tamanho das populações naturais, a despeito de seu enorme potencial de crescimento, se mantém relativamente constante ao longo do tempo, devido à limitação de recursos do ambiente.

> ***Primeira conclusão:*** em cada geração, morre um grande número de indivíduos, muitos deles sem deixar nenhum descendente.

Terceira observação: os indivíduos de uma população diferem quanto a diversas características, inclusive aquelas que influem na capacidade de explorar, com sucesso, os recursos do ambiente e de deixar descendentes.

> ***Segunda conclusão:*** os indivíduos que sobrevivem e se reproduzem a cada geração, são, provavelmente, os que apresentam mais características relacionadas com a adaptação às condições ambientais. Esta conclusão resume o conceito darwinista de seleção natural ou sobrevivência dos mais aptos.

Quarta observação: grande parte das características apresentadas por indivíduos de uma dada geração é herdada dos respectivos pais.

> ***Terceira conclusão:*** uma vez que, a cada geração, sobrevivem os mais aptos, eles tendem a transmitir aos descendentes características relacionadas a essa maior aptidão para sobreviver, isto é, para se adaptar. Em outras palavras, a seleção natural favorece, ao longo de gerações sucessivas, a permanência e o aprimoramento de características relacionadas à adaptação.

2.3 Teoria da Evolução Natural

A evolução natural é o processo que guia o surgimento de estruturas orgânicas complexas e adaptadas ao ambiente (Bäck *et al.*, 1997). De forma sucinta, e com grau elevado de simplificação, Bäck *et al.* (1997) descreve a evolução como o resultado da influência mútua entre a criação de informação genética nova, sua avaliação e seleção. Um indivíduo de uma população é afetado por outros indivíduos da população (por exemplo, pela competição por alimento, predadores, acasalamento, etc.) e também pelo ambiente em que vive (por exemplo, pela oferta de comida, clima, etc.). Quanto maior a adaptação de um indivíduo a tais condições, maior a chance do indivíduo sobreviver por mais tempo e gerar uma prole, que por sua vez herda a informação genética dos pais (possivelmente com algum erro de cópia e sujeita à recombinação das informações fornecidas pelos pais). Ao longo do processo de evolução, vai ocorrendo na população uma penetração majoritária de informação genética oriunda de indivíduos com adaptação acima da média. Além disso, o caráter não-determinístico da reprodução leva a uma produção permanente de informação genética nova e, portanto, à criação de descendentes diferenciados (para maiores detalhes veja Atmar (1994) e Fogel (1999)).

Indivíduos e espécies podem ser vistos como uma dualidade entre seu código genético (genótipo) e suas características comportamentais, fisiológicas e morfológicas (fenótipo) (Fogel, 1994). Em sistemas evoluídos naturalmente, não existe uma relação biunívoca entre um gene (elemento do genótipo) e uma característica (elemento do fenótipo): um único gene pode afetar diversos traços fenotípicos simultaneamente (pleiotropia) e uma única característica fenotípica pode ser determinada pela interação de vários genes (poligenia). Os efeitos de pleiotropia e poligenia geralmente tornam os resultados de variações genéticas imprevisíveis. Sistemas naturais em evolução são fortemente pleiotrópicos e altamente poligênicos (Hartl & Clarck, 1989). O mesmo não ocorre em sistemas artificiais, onde uma das principais preocupações é com o custo computacional do sistema, de modo que geralmente existe uma relação de um-para-um entre genótipo e fenótipo. Segundo Atmar (1994) e Fogel (1999), o processo de evolução pode ser formalizado do seguinte modo: considere dois espaços de estados distintos – um espaço de estados genotípico (codificação) G e um espaço fenotípico (comportamental) F. Considere também um alfabeto de entrada composto de símbolos provenientes do ambiente I.

O processo de evolução de uma população entre duas gerações consecutivas encontra-se esquematizado na Figura 2.1 a seguir.

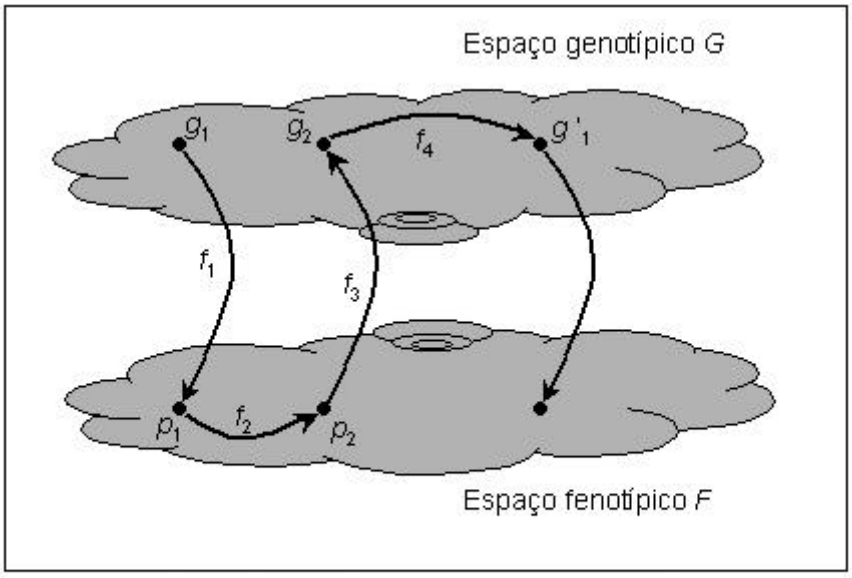


Figura 2.1 Processo Evolutivo descrito como uma seqüência de mapeamentos entre o espaço genotípico e o espaço fenotípico (figura originalmente proposta por Atmar (1994))

Existem quatro mapeamentos atuando neste processo:

$$f_1 : I \times G \rightarrow F$$

$$f_2 : F \rightarrow F$$

$$f_3 : F \rightarrow G$$

$$f_4 : G \rightarrow G$$

O mapeamento f_1, denominado *epigênese*, mapeia elementos $g_1 \in G$ em uma coleção particular de fenótipos p_1 do espaço fenotípico F, cujo desenvolvimento é modificado por seu ambiente, um conjunto de símbolos $\{i_1,...,i_k\} \in I$. Este mapeamento é inerentemente de muitos-para-um, pois pode existir uma infinidade de genótipos que acabam resultando em um mesmo fenótipo: um conjunto de códigos não expressos (não participantes na produção do fenótipo) podem existir em g_1 (Atmar, 1994).

O mapeamento f_2, *seleção*, mapeia fenótipos p_1 em p_2. Este mapeamento descreve os processos de seleção, imigração e emigração de indivíduos, além de outras influências do

meio, dentro da população local. Como a seleção natural opera apenas nas expressões fenotípicas do genótipo, o código g_1 não está diretamente envolvido no mapeamento f_2. Atmar (1994) enfatiza que a seleção atua apenas no sentido de eliminar as variantes comportamentais menos apropriadas do inevitável excesso da população, já que se assume aqui que os recursos provenientes do ambiente são limitados, exigindo a competição pela sobrevivência. Neste processo de competição, a seleção nunca opera sobre uma característica simples isoladamente do conjunto comportamental.

O mapeamento f_3, *representação* (Atmar, 1994) ou *sobrevivência genotípica* (Fogel, 1999), descreve os efeitos do mapeamento f_2 em G.

O mapeamento f_4, *mutação e recombinação*, mapeia códigos $g_2 \in G$ em $g'_1 \in G$. Este mapeamento descreve as operações de mutação e recombinação, abrangendo todas as alterações genéticas. A mutação é um erro de cópia no processo de transmissão do código genético dos pais para a sua prole. Em um universo com diferencial de entropia positivo, erros de replicação são inevitáveis e a otimização evolutiva torna-se um efeito inerente a qualquer população que se reproduz em uma arena limitada (Atmar, 1994).

Com a criação da nova população de genótipos g'_1, uma geração está completa, de maneira que a adaptação evolutiva ocorre em sucessivas iterações destes mapeamentos.

O biólogo Sewell Wright propôs, em 1931, o conceito de superfície de adaptação para descrever o nível de adaptação de indivíduos e espécies ao ambiente (Fogel, 1999). Uma população de genótipos é mapeada em seus respectivos fenótipos que, por sua vez, são mapeados nos seus valores de adaptação. Cada pico (máximo local) da superfície de adaptação corresponde a um fenótipo otimizado e, portanto, a um ou mais conjuntos de genótipos otimizados. A evolução é um processo que conduz, de forma probabilística, populações em direção a picos da superfície, enquanto que a seleção elimina variantes fenotípicas menos apropriadas. Outros pesquisadores propõem uma visão invertida da superfície de adaptação: populações avançam descendo picos da superfície de adaptação até que um ponto de mínimo seja encontrado.

Qualquer que seja o ponto de vista, a evolução é sempre um processo de otimização. Dadas as condições iniciais, restrições ambientais e parâmetros evolutivos, a seleção produzirá fenótipos tão próximos do ótimo quanto possível. Observa-se, no entanto, que em sistemas biológicos reais, não existem superfícies de adaptação estáticas, já que o ambiente está em

constante mudança, fazendo com que populações estejam em permanente evolução e co-evolução em direção a novos pontos de ótimo. Neste caso, assumindo que as mudanças ambientais são significativas, embora graduais, a taxa evolutiva deve ser suficientemente elevada para acompanhar as mudanças ambientais.

2.4 Computação Evolutiva

A computação evolutiva trata de algoritmos inspirados na teoria de evolução natural de Darwin, podendo desempenhar os seguintes papéis básicos: (a) ferramenta adaptativa para solução de problemas; (b) modelo computacional de processos evolutivos naturais.

Os sistemas inspirados em computação evolutiva mantêm uma população de soluções potenciais, aplicando processos de seleção baseados na adaptação de um indivíduo e, também, empregando outros operadores evolutivos. Diversas abordagens para sistemas baseados em evolução foram propostas, sendo que as principais diferenças entre elas dizem respeito aos operadores genéticos utilizados (uma discussão sobre operadores genéticos será apresentada mais adiante neste capítulo). As principais abordagens propostas na literatura são: (a) algoritmos genéticos; (b) estratégias evolutivas; (c) programação evolutiva.

Os a*lgoritmos genéticos* foram introduzidos por J. Holland em 1973 (Holland, 1973) com o objetivo de formalizar matematicamente e explicar rigorosamente processos de adaptação em sistemas naturais e desenvolver sistemas artificiais (simulados em computador) que retenham os mecanismos originais encontrados em sistemas naturais. Os algoritmos genéticos empregam os operadores de *crossover* e mutação (serão apresentados mais adiante neste capítulo). Os algoritmos genéticos têm sido intensamente aplicados em problemas de otimização, apesar de não ter sido este o propósito original que levou ao seu desenvolvimento.

Uma extensão dos algoritmos genéticos, denominada *programação genética*, foi introduzida por Koza (1992), em que o código genético corresponde a uma árvore de atributos, em lugar da codificação em lista de atributos comumente empregada no caso dos algoritmos genéticos. Koza detém uma patente sobre programação genética. A programação genética teve por objetivo inicial evoluir programas de computador usando os princípios da evolução natural. Atualmente, a programação genética tem sido aplicada a uma grande variedade de problemas como, por exemplo, na síntese de circuitos elétricos analógicos (Koza *et al.*, 1997) e na definição de arquiteturas de redes neurais artificiais (Gruau, 1994).

Estratégias evolutivas (Rechenberg, 1973; Schwefel, 1995) foram inicialmente propostas com o objetivo de solucionar problemas de otimização de parâmetros, tanto discretos como contínuos, empregando apenas o operador de mutação.

A *programação evolutiva*, introduzida por Fogel *et al.* (1966), foi originalmente proposta como uma técnica para criar inteligência artificial pela evolução de máquinas de estado finito (empregando, também, apenas mutação). Recentemente, tem sido aplicada a problemas de otimização, sendo, neste caso, virtualmente equivalente às estratégias evolutivas. Atualmente, existem apenas pequenas diferenças no que diz respeito aos procedimentos de seleção e codificação de indivíduos presentes nestas duas abordagens (Fogel, 1994).

Apesar das abordagens acima citadas terem sido desenvolvidas de forma independentes, seus algoritmos possuem uma estrutura comum. O termo *algoritmo evolutivo* será utilizado para denominar todos os sistemas baseados em evolução e a sua estrutura é apresentada na Figura 2.2 (Michalewicz, 1996).

Procedimento programa evolutivo
início
$t \leftarrow 0$
inicializar $P(t)$
avaliar $P(t)$
enquanto não (condição de parada) **faça**
início
$t \leftarrow t+1$
selecionar $P(t)$ a partir de $P(t-1)$
aplicar operadores genéticos a $P(t)$
avaliar $P(t)$
fim
fim

Figura 2.2 Uma proposta de um algoritmo evolutivo.

Um algoritmo evolutivo mantém uma população de indivíduos $P(t) = \left\{x_1^t,...,x_n^t\right\}$ na (geração) iteração t. Cada indivíduo representa um candidato à solução do problema em questão e, em

qualquer implementação computacional, assume a forma de alguma estrutura de dados S. Cada solução x_i^t é avaliada e produz alguma medida de adaptação ou *fitness*. Assim, uma nova população (iteração $t+1$) é formada pela seleção dos indivíduos com base nas medidas de adaptação (passo de seleção). Alguns indivíduos da população são submetidos a transformações (passo de alteração) por meio de operadores genéticos, para formar novas soluções. Existem transformações unárias m_i (mutação) que criam novos indivíduos por meio de pequenas mudanças em um indivíduo ($m_i : S \to S$) e transformações de ordem superior c_j (como o *crossover*), criando novos elementos pela combinação de dois ou mais indivíduos ($c_j : S \times ... \times S \to S$). Uma condição de parada deve ser definida (por exemplo, um determinado número de gerações) e o indivíduo da população que apresentar o melhor desempenho será tomado como a solução do problema.

Todos os algoritmos evolutivos até hoje propostos diferem em diversos aspectos pontuais: nas estruturas de dados utilizadas para codificar um indivíduo, nos operadores genéticos empregados, nos métodos para criar a população inicial, nos métodos para selecionar indivíduos para a geração seguinte, etc. Entretanto, compartilham do mesmo princípio, ou seja, uma população sofre algumas transformações aleatórias e durante a evolução os seus indivíduos competem pela sobrevivência.

2.5 Algoritmos Genéticos

Os algoritmos genéticos empregam uma terminologia originada da teoria da evolução natural e da genética. Um indivíduo da população é representado por um único *cromossomo*, o qual contém a *codificação* (genótipo) de uma possível solução do problema (fenótipo). Cromossomos são usualmente implementados na forma de vetores de atributos, onde cada elemento do vetor é denominado *gene*. Os possíveis valores que um determinado gene pode assumir são denominados *alelos*.

O processo de evolução executado por um algoritmo genético corresponde a um processo de busca em um espaço de soluções potenciais para o problema. Como enfatiza Michalewicz (1996), esta busca requer um equilíbrio entre dois objetivos aparentemente conflitantes: o aproveitamento das melhores soluções e a exploração do espaço de busca (explotação e exploração). Métodos de otimização do tipo *hillclimbing* são exemplos de métodos

que aproveitam a melhor solução na busca de possíveis aprimoramentos; em compensação, estes métodos ignoram a exploração do espaço de busca. Métodos de busca aleatória são exemplos típicos de métodos que exploram o espaço de busca, ignorando o aproveitamento de regiões promissoras do espaço. Algoritmos genéticos constituem uma classe de métodos de busca de propósito geral que apresentam um balanço notável entre aproveitamento de melhores soluções e exploração do espaço de busca.

Os algoritmos genéticos pertencem à classe dos algoritmos probabilísticos, mas eles não são métodos de busca puramente aleatórios, pois eles combinam elementos de métodos de busca diretos e estocásticos. Outra propriedade importante dos algoritmos genéticos (assim como de todos os algoritmos evolutivos) é que eles mantêm uma população de soluções candidatas enquanto que métodos alternativos, como *simulated annealing* (Aarts & Korst, 1989), processam um único ponto no espaço de busca a cada instante.

O processo de busca realizado pelos algoritmos genéticos é multi-direcional, através da manutenção de soluções candidatas, e encoraja a troca de informação entre as direções. A cada geração, soluções relativamente "boas" se reproduzem com maior freqüência que soluções relativamente "ruins". Para fazer a distinção entre diferentes soluções é empregada uma função-objetivo (de avaliação ou de adaptabilidade) que simula o papel da pressão exercida pelo ambiente sobre o indivíduo.

A estrutura de um algoritmo genético é a mesma do algoritmo evolutivo apresentado na Figura 2.2. Podemos descrever um algoritmo genético como segue (Michalewicz, 1996). Durante a iteração t, é mantida uma população de soluções potenciais (cromossomos, vetores de atributos), $P(t) = \left\{x_1^t, ..., x_n^t\right\}$. Cada solução x_i^t é avaliada e produz uma medida de sua adaptação, ou *fitness*. Uma nova população (iteração $t+1$) é então formada privilegiando a participação dos indivíduos mais adaptados. Alguns membros da nova população passam por alterações por meio de *crossover* e mutação para formar novas soluções potenciais. Este processo se repete até que um número pré-determinado de iterações seja atingido, ou até que o nível de adaptação esperado seja alcançado pelo melhor indivíduo da população.

Em resumo, um algoritmo genético para um problema particular deve ter os seguintes componentes:

- uma representação genética para soluções candidatas ou potenciais (processo de codificação);
- uma maneira de criar uma população inicial de soluções candidatas ou potenciais;

- uma função de avaliação que faz o papel da pressão ambiental, classificando as soluções em termos de sua adaptação ao ambiente (ou seja, sua capacidade de resolver o problema);
- operadores genéticos;
- valores para os diversos parâmetros usados pelo algoritmo genético (tamanho da população, probabilidades de aplicação dos operadores genéticos, etc.);
- Operadores de seleção.

2.5.1 Codificação de indivíduos: representação genética

Cada indivíduo de uma população representa um candidato em potencial à solução do problema em questão. No algoritmo genético clássico, proposto por Holland (1973; 1992) as soluções candidatas são codificadas em arranjos binários de tamanho fixo. A motivação para o uso de codificação binária vem da teoria dos esquemas (*schemata theory*), descrita por Holland (1992). Holland (1992) argumenta que seria benéfico para o desempenho do algoritmo maximizar o paralelismo implícito inerente ao algoritmo genético e prova que um alfabeto binário maximiza o paralelismo implícito.

Entretanto, em diversas aplicações práticas a utilização de codificação binária leva a um desempenho insatisfatório. Em problemas de otimização numérica com parâmetros reais, algoritmos genéticos com representação em ponto flutuante freqüentemente apresentam desempenho superior à codificação binária. Michalewicz (1996) argumenta que a representação binária apresenta desempenho pobre quando aplicada a problemas numéricos com alta dimensionalidade e onde alta precisão é requerida. Suponha por exemplo, que temos um problema com 100 variáveis com domínio no intervalo [-500, 500] e que precisamos de seis dígitos decimais de precisão após vírgula. Com isso, em uma representação binária de ponto fixo, seriam necessários dez bits para a parte inteira com sinal e 20 bits para a parte fracionária de cada variável. Neste caso precisaríamos de um cromossomo com 3000 bits para representar as 100 variáveis, e teríamos um espaço de busca de dimensão $2^{3000} \cong 10^{1000}$. Neste tipo de problema, o algoritmo genético clássico apresenta desempenho pobre. Michalewicz (1996) apresenta também simulações computacionais comparando o desempenho de algoritmos genéticos com codificação binária e com ponto flutuante, aplicados a um problema de controle.

Os resultados apresentados mostram uma clara superioridade da codificação em ponto flutuante para este problema.

A argumentação de Michalewicz (1996), de que o desempenho de um algoritmo genético com codificação binária é pobre quando o espaço de busca é de dimensão elevada, não é universalmente aceita na literatura referente a algoritmos genéticos. Fogel (1994) argumenta que o espaço de busca por si só (sem levar em conta a escolha da representação) não determina a eficiência do algoritmo genético. Espaços de busca de dimensão elevada podem às vezes ser explorados eficientemente, enquanto que espaços de busca de dimensão reduzida podem apresentar dificuldades significativas. Fogel (1994), entretanto, concorda que a maximização do paralelismo implícito nem sempre produz um desempenho ótimo.

Fica claro, portanto, que a codificação é uma das etapas mais críticas na definição de um algoritmo genético. A definição inadequada da codificação pode levar a problemas de convergência prematura do algoritmo genético, dentre outros efeitos indesejáveis.

Em problemas de otimização restrita, por sua vez, existe uma problemática adicional, pois a codificação adotada pode fazer com que indivíduos modificados por *crossover*/mutação sejam inválidos. Nestes casos, cuidados especiais devem ser tomados na definição da codificação e/ou dos operadores (Bäck *et al.,* 1997).

2.5.2 Formação da população inicial

O método mais comum utilizado na criação da população é a inicialização aleatória dos indivíduos. Se algum conhecimento inicial a respeito do problema estiver disponível, pode ser utilizado na inicialização da população. Por exemplo, se é sabido que a solução final (assumindo codificação binária) vai apresentar mais 0's do que 1's, então esta informação pode ser utilizada, mesmo que não se saiba exatamente a proporção. Já em problemas com restrição, deve-se tomar cuidado para não gerar indivíduos inválidos na etapa de inicialização (Bäck *et al.,* 1997).

2.5.3 Função de avaliação

Bäck *et al.* (1997) nos explica que os algoritmos genéticos requerem uma forma de mapear a representação dos cromossomos (genótipo) em um valor numérico associado ao nível de adaptação ou *fitness*. Este mapeamento pode ou não requerer a produção explícita de

fenótipo do indivíduo. Os operadores de recombinação (*crossover* e mutação) trabalham diretamente na representação codificada e não no fenótipo (veja Figura 2.1). Para formalizar este conceito, suponha que a função-objetivo seja dada na forma:

$$f : M \to \Re$$

representando um mapeamento do espaço das soluções decodificadas M para o eixo dos reais $\Re$. Esta função de avaliação ou *fitness* F é descrita como:

$$F : R \xrightarrow{d} M \xrightarrow{f} \Re \xrightarrow{s} \Re_+$$

$$F = s \circ f \circ d$$

onde R é o espaço da representação do cromossomo, d é uma função de decodificação, e s é uma função de normalização. A função de normalização s é freqüentemente utilizada em combinação com "seleção proporcional", de modo a garantir que os valores do *fitness* serão sempre de mesmo sinal e também para garantir a maximização do *fitness*. Por exemplo, quando uma função-objetivo f, que mapeia n valores reais em $\Re$, é codificada usando representação binária, a função de *fitness* F pode ser escrita como:

$$F : \{0,1\}^l * \{0,1\}^l * \ldots * \{0,1\}^l \xrightarrow{d} \Re^n \xrightarrow{f} \Re \xrightarrow{s} \Re_+$$

onde l é o numero de bits da representação real (Bäck *et al.,* 1997).

Bäck *et al.* (1997) também nos ensina que, em contrapartida aos algoritmos genéticos, as "estratégias evolutivas" e a "programação evolutiva" trabalham diretamente no segundo espaço M. Desta forma, eles não requerem funções de decodificação d. Eles tipicamente também não precisam da função de normalização s. Com isto a avaliação do *fitness* é completamente especificada pela equação

$$f : M \to \Re$$

No entanto, a determinação absoluta e precisa do *fitness* pode não ser elementar. A dificuldade vem da objetividade da função de *fitness*, a qual freqüentemente é criada somente quando se possui conhecimento significativo sobre o espaço de busca (Angelini, 1993). Para eliminar a dependência de funções de *fitness* objetivas, uma competição é introduzida. Função de *fitness* competitivo é um método para calcular *fitness* que é dependente da população atual, enquanto que as funções de *fitness* padrão retornam o mesmo valor de *fitness* para um indivíduo independente dos outros membros desta população. A vantagem da competição é que os algoritmos evolucionários não necessitam de valores precisos de *fitness*, porque muitos esquemas de seleção trabalham apenas comparando os valores de *fitness*; ou seja, os critérios

"melhores ou piores". Em outras palavras, não é necessário o valor da medida absoluta de *fitness*, mas sua medida relativa deve ser derivada quando um indivíduo é confrontado com outros indivíduos. Este método mostra-se bastante adequado para implementações usando paralelismo (Bäck *et al.,* 1997).

2.5.4 Operadores genéticos

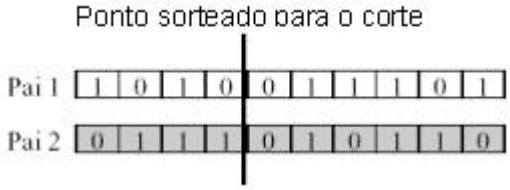
Os operadores genéticos mais freqüentemente utilizados são o *crossover* e a mutação. Nesta seção, são apresentados os principais aspectos relacionados a estes operadores.

O operador de *Crossover*

O operador de *crossover* ou recombinação cria novos indivíduos por intermédio da combinação de dois ou mais indivíduos. A idéia intuitiva por trás do operador de *crossover* é a troca de informação entre diferentes soluções candidatas. No algoritmo genético clássico, é atribuída uma probabilidade de *crossover* fixa aos indivíduos da população.

O operador de *crossover* mais comumente empregado é o *crossover* de um ponto. Para a aplicação desse operador, são selecionados dois indivíduos (pais) e a partir de seus cromossomos são gerados dois novos indivíduos (filhos). Para gerar os filhos, seleciona-se um ponto de corte aleatoriamente nos cromossomos-pais, de modo que os segmentos a partir do ponto de corte sejam trocados. Veja o exemplo a seguir:

Exemplo 1: Considere dois indivíduos selecionados como pais, a partir da população da geração atual de um algoritmo genético, e suponha que o ponto de corte escolhido (aleatoriamente) encontra-se entre as posições 4 e 5 dos cromossomos-pais:



Após o *crossover*, são gerados os seguintes cromossomos-filhos:

Outros tipos de *crossover* têm sido propostos na literatura. Uma extensão simples do *crossover* de um ponto é o *crossover* de dois pontos. Neste, dois pontos de corte são escolhidos e o material genético será invertido entre eles na posição de ruptura. Observe o exemplo a seguir:

Exemplo 2: Considere os mesmos cromossomos-pais do Exemplo 1. Suponha que os pontos de *crossover* escolhidos estão localizados entre as posições 3 e 4 e entre as posições 7 e 8. Assim, os novos indivíduos produzidos serão:

Outro tipo de *crossover* muito comum é o *crossover uniforme* (Syswerda, 1989): para cada bit no primeiro filho é decidido (com alguma probabilidade fixa *p*) qual pai vai contribuir com seu respectivo bit para aquela posição. Considere o seguinte exemplo:

Exemplo 3: Tomando como base os mesmos cromossomos-pais dos exemplos anteriores, seja $p = 0,5$. Podem ser obtidos pela aplicação do *crossover* uniforme os seguintes cromossomos-filhos:

Como o *crossover* uniforme troca bits ao invés de segmentos de bits (que aqui fazem o papel dos genes), ele pode combinar características independentemente da sua posição relativa no cromossomo.

Eshelman *et al.* (1989) relata diversos experimentos com vários operadores de *crossover*. Os resultados indicam que o operador com pior desempenho é o *crossover* de um ponto; entretanto, não há nenhum operador de *crossover* que apresente um desempenho superior aos demais em todos os casos. Uma conclusão a que se pode chegar a partir desses resultados é que cada operador de *crossover* é particularmente eficiente para um determinado conjunto de

problemas e pode ser extremamente ineficiente para outros. Uma abordagem sistemática para definição prévia do melhor operador de *crossover* para cada caso representa ainda um desafio para os pesquisadores que atuam em computação evolutiva.

Apesar disto, embora não seja uma razão determinística para explicar o diferencial de desempenho entre os vários tipos de operadores de *crossover*, é fato que, no caso de a ordenação adotada para posicionar os genes ao longo do cromossomo não interferir no fenótipo, o *crossover* uniforme tende a apresentar um desempenho superior, por ser justamente aquele que não leva em conta a ordenação dos genes, já que opera cada um individualmente. É evidente que um dado gene em um cromossomo, quando selecionado para *crossover* uniforme, será trocado pelo gene de mesma posição no outro cromossomo (aqui a ordem importa), mas esta alteração não interfere na probabilidade de troca dos genes vizinhos (é aqui que a ordem não importa).

Os operadores de *crossover* descritos até aqui também podem ser utilizados com codificação em ponto flutuante, embora existam operadores de recombinação especialmente desenvolvidos para estes casos. Um exemplo específico para o caso de codificação em ponto flutuante é o chamado *crossover aritmético* (Michalewicz, 1996). Este operador é definido como uma fusão de dois vetores (cromossomos): se x_1 e x_2 são dois indivíduos selecionados para *crossover*, os dois filhos resultantes serão $x'_1 = ax_1 + (1-a)x_2$ e $x'_2 = (1-a)x_1 + ax_2$, sendo a um número aleatório pertencente ao intervalo [0, 1]. Esse operador é particularmente apropriado para problemas de otimização numérica com restrições, onde a região factível é convexa. Se x_1 e x_2 pertencem à região factível, combinações convexas de x_1 e x_2 serão também factíveis, garantindo que o *crossover* não vai gerar indivíduos inválidos para o problema em questão. Outros exemplos de *crossover* especialmente desenvolvidos para utilização em problemas de otimização numérica restritos e codificação em ponto flutuante são o *crossover geométrico* e o *crossover esférico*, descritos em Michalewicz & Schoenauer (1996).

Um aspecto importante em um algoritmo genético diz respeito a como escolher os indivíduos que serão submetidos à recombinação. Aqui também existem diversas alternativas possíveis, sendo que entre as mais comuns destacam-se:

- *crossover* entre indivíduos seguindo alguma distribuição de probabilidade: são escolhidos indivíduos da população atual aleatoriamente ou por meio de *Roulette Wheel* (veja Seção 2.5.6- Técnicas de seleção);

- *crossover* entre um indivíduo aleatório e o melhor indivíduo: é escolhido um indivíduo da população atual aleatoriamente ou por meio de *Roulette Wheel*, sendo o outro elemento o melhor da população.

O Operador de Mutação

O operador de mutação modifica aleatoriamente um ou mais genes de um cromossomo. A probabilidade de ocorrência de mutação em um gene é denominada *taxa de mutação*. Usualmente, são atribuídos valores pequenos para a taxa de mutação. A idéia intuitiva por trás desse operador é a criação de variabilidade extra na população, mas sem destruir o progresso já obtido no decorrer do processo evolutivo, ou seja, a variabilidade deve se comportar como uma perturbação de efeito localizado.

Considerando codificação binária, o operador de mutação padrão simplesmente troca o valor do gene selecionado (Holland, 1992). Assim, se um gene selecionado para mutação tiver valor 1, passará para 0 após a aplicação do operador, e vice-versa.

No caso de problemas com codificação em ponto flutuante, o operador de mutação mais popular é a *mutação gaussiana* (Michalewicz & Schoenauer, 1996), modificando todos os componentes de um cromossomo $x = [x_1 ... x_n]$ na forma:

$$x' = x + N(0,\sigma),$$

sendo $N(0,\sigma)$ um vetor de variáveis aleatórias independentes, com distribuição normal, média zero e desvio padrão σ.

Um operador importante para problemas em que os indivíduos empregam codificação em ponto flutuante é a *mutação uniforme* (Michalewicz, 1996). Este operador seleciona aleatoriamente um componente $k \in \{1,2,...,n\}$ do cromossomo $x = [x_1 ... x_k ... x_n]$ e gera um indivíduo $x' = [x_1 ... x_k' ... x_n]$, onde x_k' é um número aleatório (com distribuição de probabilidade uniforme) amostrado no intervalo [*LB*, *UB*]. *LB* e *UB* são, respectivamente, os limites inferior e superior da variável x_k.

Outro operador de mutação, especialmente desenvolvido para problemas de otimização com restrição e codificação em ponto flutuante é a chamada *mutação não-uniforme* (Michalewicz, 1996; Michalewicz & Schoenauer, 1996). A mutação não-uniforme é um operador dinâmico, destinado a melhorar a sintonia fina ao longo do processo evolutivo. Pode ser definido da seguinte forma: seja $x = [x_1 ... x_n]$ um cromossomo na geração *t* e suponha que o

elemento x_k ($k \in \{1,2,...,n\}$) foi selecionado para mutação. O cromossomo resultante será $x' = [x_1 ... x_k' ... x_n]$, com

$$x_k' = \begin{cases} x_k + \Delta(t, UB - x_k) & \text{, com 50\% de probabilidade} \\ x_k - \Delta(t, x_k - LB) & \text{, com 50\% de probabilidade} \end{cases}$$

A função $\Delta(t, y)$ retorna um valor no intervalo $[0, y]$, tal que a probabilidade de $\Delta(t, y)$ ser próximo de zero aumenta à medida que t aumenta. Esta propriedade faz com que este operador explore mais amplamente o espaço nas gerações iniciais (quando t é pequeno) e mais localmente em gerações avançadas (quando t é grande). Michalewicz (1996) propõe a seguinte função, sendo r um número aleatório no intervalo [0, 1], T um número máximo de gerações e b um parâmetro que determina o grau de dependência do número de iterações (valor proposto: b = 5):

$$\Delta(t, y) = y.(1 - r^{(1-t/T)^b})$$

Este operador foi usado com sucesso por de Castro *et al.* (1998) na evolução de condições iniciais para o treinamento de redes neurais artificiais. Outros exemplos de operadores de mutação para problemas de otimização numérica e com codificação em ponto flutuante podem ser encontrados em Michalewicz & Schoenauer (1996).

2.5.5 Valores para os diversos parâmetros

Apesar do desempenho de um algoritmo genético depender da escolha adequada de seus parâmetros, não existem critérios sistemáticos e genéricos que estabeleçam valores para alguns parâmetros críticos em algoritmos genéticos, tais como o tamanho da população, os mecanismos de seleção, e os valores de taxas de *crossover* e mutação (Guerrero *et. al.*, 1999).

Taxas de *Crossover* e Mutação

Os algoritmos evolutivos clássicos exigem definição por parte do usuário dos diversos parâmetros descritos anteriormente, como tamanho da população e probabilidades de *crossover* e mutação. Estes parâmetros permanecem fixos durante toda a execução do algoritmo ou são alterados arbitrariamente em pontos específicos do processo evolutivo. Em geral, os valores assumidos por estes parâmetros são determinantes para que o algoritmo seja capaz de encontrar uma solução de qualidade, e também para a eficiência na busca desta

solução. Entretanto, encontrar valores apropriados para estes parâmetros é uma tarefa custosa em termos computacionais, pois não há uma metodologia eficiente que ajude nesta definição. Assim, muitas abordagens têm sido propostas no sentido do ajuste destes valores durante a execução do algoritmo. Em Eiben *et al*. (1999), encontram-se diversas técnicas para o controle destes parâmetros durante a execução do algoritmo, mostrando-se uma das áreas de pesquisa mais promissoras em computação evolutiva. Como um exemplo mais voltado para o problema de encontrar uma boa árvore filogenética veja Skourikhine (2000) .

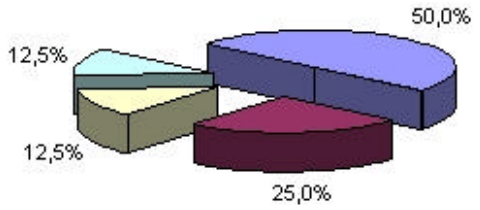
Sistemas Co-Evolutivos

Em sistemas co-evolutivos, conforme descrito por Jan (1995), mais de um processo evolutivo ocorre simultaneamente. Usualmente, consideram-se mais de uma população (por exemplo, uma população de “presas” e outra de “predadores”) como parte de um processo interativo. Em sistemas desse tipo, a função de *fitness* de uma população pode depender do estado da outra população (Potter & DeJong, 2000). Sistemas co-evolutivos podem ser abordagens interessantes para problemas de larga escala, de modo que um problema complexo é decomposto em subproblemas menores. Dessa maneira, existiria um processo evolutivo para cada subproblema e os processos estariam todos inter-relacionados. Esta parece ser uma proposta interessante para ser aplicada na reconstrução de árvores filogenéticas, onde o espaço de solução tende a ser muito grande, como pode ser observado no Apêndice B.

2.5.6 Técnicas de seleção

O algoritmo genético clássico utiliza um esquema de seleção de indivíduos para a próxima geração chamado *Roulette Wheel* (Goldberg, 1989). O *Roulette Wheel* atribui a cada indivíduo de uma população uma probabilidade de passar para a próxima geração proporcional ao seu *fitness* medido, em relação à somatória do *fitness* de todos os elementos da população. Assim, quanto maior for o *fitness* de um indivíduo, maior a probabilidade dele passar para a próxima geração. Na Figura a seguir, pode ser observado um exemplo de aplicação do *Roulette Wheel* onde cada indivíduo em uma determinada geração recebe uma probabilidade de passar à próxima geração proporcional ao seu *fitness*, medido em relação ao *fitness* total da população.

Nº	Comossomo	*Fitness*	Graus	% do Total
1	0001110101	6,0	180	50,0
2	0101110011	3,0	90	25,0
3	1101010001	1,5	45	12,5
4	1101010011	1,5	45	12,5
Total		**12,0**	**360**	**100,0**



Como pode ser observado, a seleção de indivíduos por *Roulette Wheel* pode fazer com que o melhor indivíduo da população seja perdido, já que a probabilidade dele ser escolhido para compor a próxima geração, apesar de ser alta, é menor que 1. Um recurso paliativo seria escolher como solução o melhor indivíduo encontrado ao longo de todas as gerações do algoritmo. Outra opção mais consistente é simplesmente manter sempre o melhor indivíduo da geração atual na geração seguinte, estratégia esta conhecida como *seleção elitista* (Fogel, 1994; Michalewicz, 1996). Com isso, além do melhor indivíduo ser sempre preservado, ele vai continuar a contribuir com seu código genético na produção de descendentes que irão compor as próximas gerações.

Outro exemplo de mecanismo de seleção é a *seleção baseada em rank* (Bäck *et al.,* 1997). Esta estratégia utiliza as posições dos indivíduos quando ordenados de acordo com o *fitness* para determinar a probabilidade de seleção, podendo ser usados mapeamentos lineares ou não-lineares para determinar esta probabilidade.

Um exemplo de mapeamento não-linear pode ser visto em Michalewicz (1996). Uma variação deste mecanismo é simplesmente passar os *N* melhores indivíduos para a próxima geração.

A seguir, são apresentados outros possíveis mecanismos de seleção. É importante deixar claro que estes mecanismos de seleção operam sobre os indivíduos da população intermediária da geração atual, com o objetivo de formar a população da próxima geração:

- seleção por diversidade: são selecionados os indivíduos mais diversos da população;
- seleção bi-classista: são selecionados os *P*% melhores indivíduos e os (100 - *P*)% piores indivíduos;

2.6 Diversidade populacional e deriva genética

Syed (1995) apresenta um interessante estudo sobre o efeito da taxa de mutação sobre a diversidade populacional.

Ausência de pressão seletiva e ausência de mutação

Inicialmente supõe-se que cada indivíduo da população é dado por um cromossomo haplóide e que o *crossover* é uniforme. Cada gene será representado por um bit, não há mutação e não há pressão seletiva. Os resultados então obtidos serão estendidos para o caso de presença de mutação e, em seguida, presença de pressão seletiva. O estudo vai se concentrar na diversidade de um único gene dentro da população (Syed ,1995).

Como não está sendo considerada pressão seletiva e como o *crossover* é uniforme, esta análise é diretamente extensível a qualquer gene da cadeia cromossômica. O primeiro passo é definir o que se entende por freqüência de um gene numa população. A freqüência p de ocorrência do alelo 1, correspondente àquele gene, é dada por

$$p = \frac{W}{N}$$

onde N é o número total de indivíduos na população e W é o número de indivíduos que têm o alelo 1. Logo, a freqüência q de ocorrência do alelo 0, correspondente àquele gene, é dada por

$$q = 1 - p$$

A partir da definição da freqüência gênica, é possível estabelecer uma medida de diversidade deste gene na população (Syed ,1995):

$$d(p) = 1 - |2p - 1|$$

Para $p = 0$ ou $p = 1$, resulta $d(p) = 0$, o que implica que todos os indivíduos apresentam o mesmo alelo. Para $p = \frac{1}{2}$, resulta $d(p) = 1$, ou seja, o máximo nível de diversidade ocorre quando metade da população apresenta o alelo 1, e a outra metade, o alelo 0. Como não há pressão seletiva, e supondo também ausência de mutação, a próxima geração é obtida pela escolha aleatória de dois pais pertencentes à geração atual, e o *crossover* uniforme é empregado para produzir os filhos (Syed ,1995).

Dado que p é a freqüência de ocorrência do alelo 1, correspondente a um dado gene da população, a probabilidade de um filho também apresentar este alelo 1 é dada por:

$$p_f = p^2 + \frac{1}{2}p(1-p) + \frac{1}{2}(1-p)p \Rightarrow p_f = p$$

Isto implica que a probabilidade de um filho apresentar um alelo 1 é igual à probabilidade de selecionar um indivíduo da população atual que apresente um alelo 1. Com isso, para efeito da análise em questão, é possível simplificar o processo de criação da próxima geração de indivíduos pela simples seleção de N indivíduos da geração atual (Syed ,1995).

Dada uma geração k, e assumindo que p_k é a freqüência de ocorrência do alelo 1 correspondente a um dado gene da população, a probabilidade da freqüência ser $p_{k+1} = \frac{W_{w+1}}{N}$ na próxima geração é dada por (Syed ,1995):

$$P\langle p_{k+1} / p_k \rangle = \left(\begin{matrix} N \\ W_{k+1} \end{matrix} \right) p_k{}^{W_{k+1}} (1-p_k)^{N-W_{k+1}}$$

Para uma população de tamanho N, as probabilidades $P\langle p_{k+1} / p_k \rangle$ podem ser utilizadas para gerar uma matriz M de dimensão $(N+1)x(N+1)$, cujos elementos são dados por (Syed ,1995):

$$m_{ij} = P\left\langle \frac{i}{N} \middle| \frac{j}{N} \right\rangle,\ i, j = 0,1,2,...,N$$

Para a *k*-ésima geração, seja g_k um vetor-coluna de distribuição de probabilidade, de dimensão $(N+1)$, e cujo *i*-ésimo elemento é a probabilidade de que a freqüência gênica seja $\frac{i}{N}$. Se g_0 é a distribuição de probabilidade para a freqüência gênica na geração inicial, então, após *k* gerações, a distribuição de probabilidade para a freqüência gênica é dada por:

$$g_k = M^k g_0$$

Para garantir o maior nível possível de diversidade para a população inicial, g_0 pode ser definido na forma:

$$\begin{cases} g_{oi} = 0, \quad i = 0,1,2,...,N,\ i \neq j \\ g_{0j} = \begin{cases} 1, \ j = \dfrac{N}{2} \ para N \ par \\ \dfrac{1}{2}, \ j = \dfrac{N-1}{2}, \ \dfrac{N+1}{2}, \ para N \ impar \end{cases} \end{cases}$$

Por exemplo, para $N = 2$, temos

$$M = \begin{bmatrix} P\langle 0 \mid 0\rangle & P\langle 0 \mid \frac{1}{2}\rangle & P\langle 0 \mid 1\rangle \\ P\langle \frac{1}{2} \mid 0\rangle & P\langle \frac{1}{2} \mid \frac{1}{2}\rangle & P\langle \frac{1}{2} \mid 1\rangle \\ P\langle 1 \mid 0\rangle & P\langle 1 \mid \frac{1}{2}\rangle & P\langle 1 \mid 1\rangle \end{bmatrix}$$

$$g_1 = Mg_0 = \begin{bmatrix} P\langle 0 \mid 0\rangle & P\langle 0 \mid \frac{1}{2}\rangle & P\langle 0 \mid 1\rangle \\ P\langle \frac{1}{2} \mid 0\rangle & P\langle \frac{1}{2} \mid \frac{1}{2}\rangle & P\langle \frac{1}{2} \mid 1\rangle \\ P\langle 1 \mid 0\rangle & P\langle 1 \mid \frac{1}{2}\rangle & P\langle 1 \mid 1\rangle \end{bmatrix} . \begin{bmatrix} g_0(0) \\ g_0(\frac{1}{2}) \\ g_0(1) \end{bmatrix}$$

$$g_1 = \begin{bmatrix} g_1(0) \\ g_1(\frac{1}{2}) \\ g_1(1) \end{bmatrix} = \begin{bmatrix} P\langle 0 \mid 0\rangle g_0(0) + P\langle 0 \mid \frac{1}{2}\rangle g_0(\frac{1}{2}) + P\langle 0 \mid 1\rangle g_0(1) \\ P\langle \frac{1}{2} \mid 0\rangle g_0(0) + P\langle \frac{1}{2} \mid \frac{1}{2}\rangle g_0(\frac{1}{2}) + P\langle \frac{1}{2} \mid 1\rangle g_0(1) \\ P\langle 1 \mid 0\rangle g_0(0) + P\langle 1 \mid \frac{1}{2}\rangle g_0(\frac{1}{2}) + P\langle 1 \mid 1\rangle g_0(1) \end{bmatrix}$$

A seguir, é apresentado o primeiro elemento do vetor $g_2 = Mg_1 = M^2 g_0$, mosrando explicitamente o efeito das várias probabilidades condicionais sobre a distribuição de probabilidade inicial.

$$g_2(0) = P\langle 0 \mid 0\rangle P\langle 0 \mid 0\rangle g_0(0) + P\langle 0 \mid \frac{1}{2}\rangle P\langle \frac{1}{2} \mid 0\rangle g_0(0) + P\langle 0 \mid 1\rangle P\langle 0 \mid 1\rangle g_0(0) +$$

$$+ P\langle 0 \mid 0\rangle P\langle 0 \mid \frac{1}{2}\rangle g_0(\frac{1}{2}) + P\langle 0 \mid \frac{1}{2}\rangle P\langle \frac{1}{2} \mid \frac{1}{2}\rangle g_0(\frac{1}{2}) + P\langle 0 \mid 1\rangle P\langle 1 \mid \frac{1}{2}\rangle g_0(\frac{1}{2}) +$$

$$+ P\langle 0 \mid 0\rangle P\langle 0 \mid 1\rangle g_0(1) + P\langle 0 \mid \frac{1}{2}\rangle P\langle \frac{1}{2} \mid 1\rangle g_0(1) + P\langle 0 \mid 1\rangle P\langle 1 \mid 1\rangle g_0(1)$$

A diversidade da população após *k* gerações, dado que a distribuição de probabilidade inicial é g_0, pode ser encontrada na forma:

$$D(k) = \sum_{i=0}^{N} g_{ki} d\left(\frac{i}{N}\right) = \sum_{i=0}^{N} g_{ki} \left(1 - \left|2\frac{i}{N} - 1\right|\right)$$

Tomando valores diferentes para o tamanho N da população, o gráfico da evolução da diversidade D em função das gerações é apresentado a seguir (Syed ,1995).

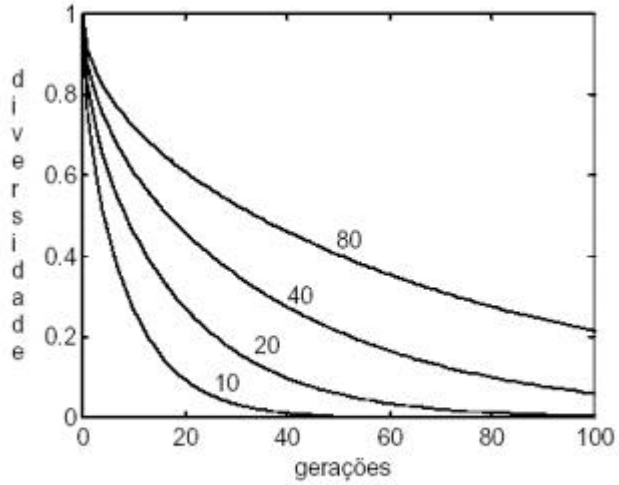


Figura 2.3 Relação entre a diversidade e o tamanho da população (Syed ,1995).

Observa-se neste gráfico que a diversidade decresce mais rapidamente para tamanhos menores de população.

O gráfico a seguir mostra mais precisamente que a taxa de decrescimento da diversidade é inversamente proporcional ao tamanho da população. No eixo x temos agora $\frac{gerações}{tamanho\ \ da\ \ população}$ para os casos de $N = 10, 20, 40$ e 80 (Syed ,1995).

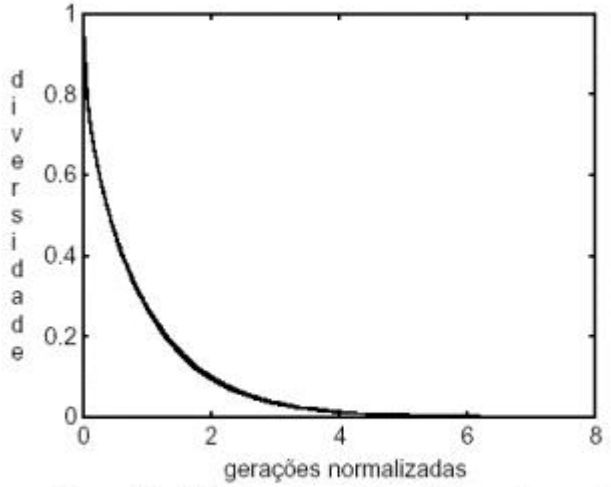


Figura 2.4 Relação entre a diversidade e o tamanho da população normalizada (Syed ,1995).

De acordo com este último gráfico, para qualquer tamanho de população, a diversidade se anula em aproximadamente 6N gerações. Observe que a perda de diversidade ocorre mesmo na ausência de pressão seletiva. Este fenômeno é conhecido como deriva genética (*genetic drift*).

Ausência de pressão seletiva, mas presença de mutação

Com a introdução do efeito de mutação, é necessário rever a construção da matriz de distribuição de probabilidade M. A taxa de mutação é definida como a probabilidade μ com que qualquer bit do recém criado indivíduo mude de 0 para 1, ou vice-versa. Assim, quando a mutação está presente, a probabilidade de que um indivíduo da próxima geração tenha um alelo 1, correspondente a um dado gene da população, é dada por (Syed ,1995):

$$p_f = p + q\mu - p\mu = p - 2p\mu + \mu$$

Dada uma geração *k*, e assumindo que p_k é a freqüência de ocorrência do alelo 1, correspondente a um dado gene da população, a probabilidade da freqüência ser $p_{k+1} = \frac{W_{w+1}}{N}$ na próxima geração é agora dada por:

$$P\langle p_{k+1} / p_k \rangle = \binom{N}{W_{k+1}} (p_k - 2p_k\mu + \mu)^{W_{k+1}} (1 - p_k + 2p_k\mu - \mu)^{N - W_{k+1}}$$

Esta formulação modificada para a distribuição de probabilidade pode agora ser utilizada para construir a matriz M e calcular a diversidade em função das gerações, como feito anteriormente. Usando $N =$ 20 para o tamanho da população e adotando diferentes taxas de mutação, o gráfico a seguir mostra o comportamento da diversidade para cada caso (Syed ,1995).

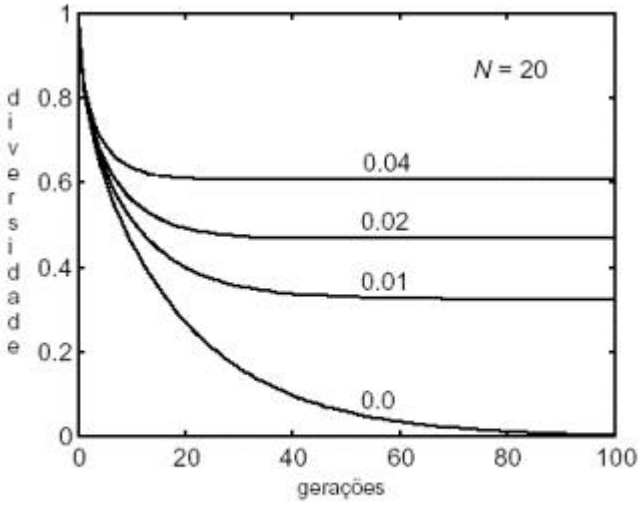


Figura 2.5 Relação entre a diversidade e o tamanho da população com presença de mutação (Syed ,1995).

Pode-se concluir que a introdução de mutação (não importa a que taxa) provoca uma desaceleração da queda da diversidade e finalmente uma estabilização em um nível de diversidade mínima. Taxas de mutação maiores resultam em níveis maiores de diversidade mínima. O gráfico a seguir mostra, para tamanhos distintos de população, como uma mesma taxa de mutação influencia o nível de diversidade (Syed ,1995).

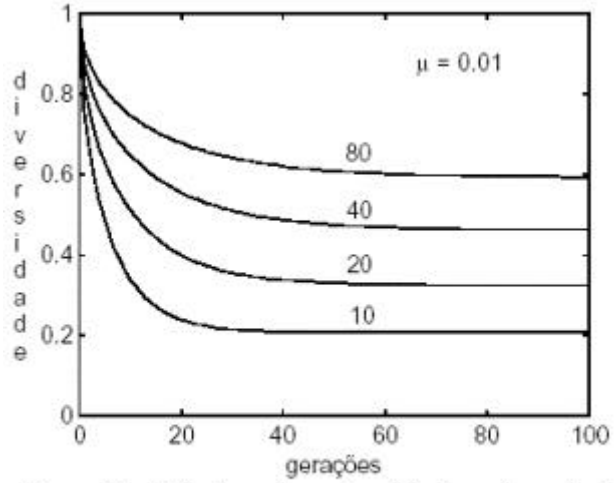


Figura 2.6 Relação entre a diversidade e o tamanho da população com taxa de mutação constante (Syed ,1995).

Pode-se observar que uma mesma taxa de mutação conduz a níveis de diversidade mínima maiores para populações maiores. Portanto, dada a taxa de mutação e o tamanho da população é possível calcular o nível de diversidade para o qual a população vai convergir (Syed ,1995).

Presença de pressão seletiva: efeito na diversidade

A introdução de pressão seletiva provoca invariavelmente um aumento da taxa de decrescimento da diversidade, ocasionando possivelmente um nível mais baixo para a diversidade mínima (Syed ,1995).

Os efeitos da pressão seletiva são produzidos por dois fatores completamente independentes (Syed ,1995):

1. redução do tamanho efetivo da população pois nem todos os indivíduos irão gerar descendentes e alguns indivíduos com maior nível de adaptação vão gerar vários descendentes;
2. tendência de que um alelo em particular de um dado gene seja favorecido pelo processo de seleção, baseado na função de adaptação (*fitness*).

2.7 Representação em Árvore e Operadores Genéticos Associados

Como pode ser observado em Foster (2001), para o caso de encontrar a árvore com maior probabilidade de reconstruir a filogenia de um conjunto de seqüências de DNAs, os cromossomos desta população deveriam ser estruturas de dados com ramificações (árvores) e não as tradicionais estruturas lineares usualmente empregadas (listas).

Conforme anteriormente exposto na seção 2.5.4 (Operadores genéticos) e em Foster (2001), a recombinação em algoritmos genéticos ocorre com a seleção e troca de um subconjunto de genes de dois indivíduos, geralmente isolando uma seqüência de cada e trocando entre eles os fragmentos resultantes, de maneira a formar dois novos indivíduos. Esta técnica é bastante simples e intuitiva para estruturas lineares mas quando usamos estruturas de dados mais complexas para sua representação, este mecanismo não permanece tão simples.

Para cromossomos na forma de árvore, é necessário criar operadores especiais para mutação e recombinação (Foster, 2001).

Conforme Manber (1989), uma árvore pode ser considerada como sendo um grafo dirigido acíclico e assim podemos representar uma árvore com uma matriz de adjacências. Vide Apêndice C para maiores detalhes.

Adotando a representação na forma de matriz de adjacências para as árvores estudadas, podemos propor três tipos de operadores de mutação:

(a) um para permutação entre folhas;

(b) outro para permutação entre ramos e

(c) outro para permutação de ramos com folhas.

Para ajudar a melhorar o entendimento sobre como os três operadores propostos funcionam, considere a árvore e sua respectiva representação em forma de matriz de adjacências das Figuras 2.7 e 2.8 a seguir.

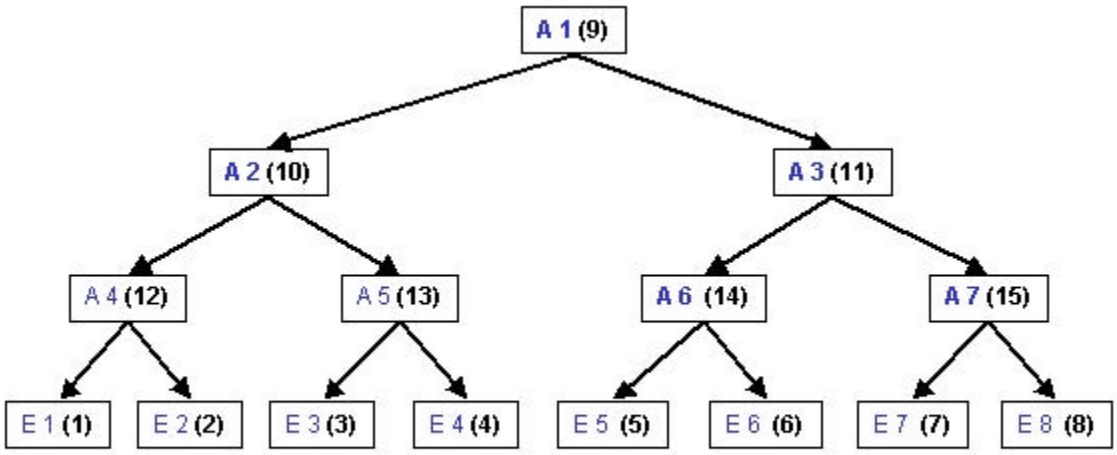


Figura 2.7 Uma possível representação gráfica da árvore filogenética de oito espécies, onde "E n" significa "espécie n" e "A n" significa "Ancestral n".

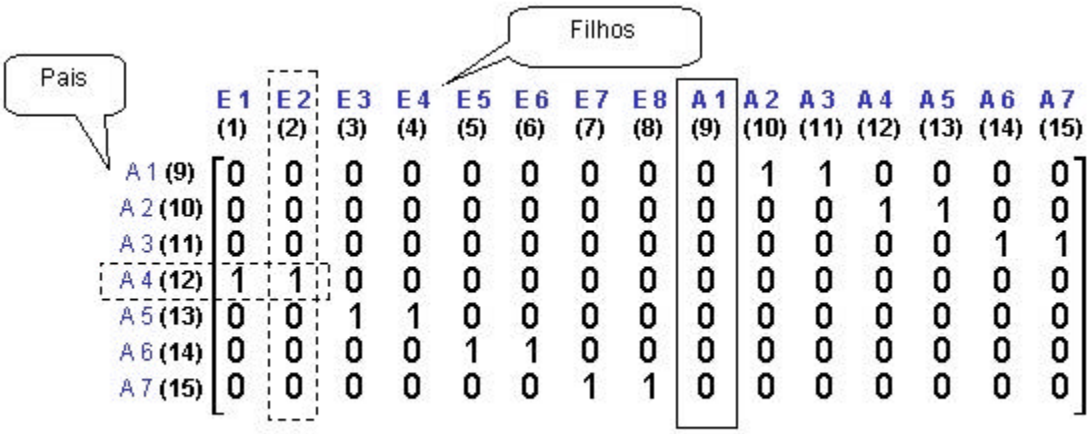


Figura 2.8 Representação na forma de matriz de adjacências da árvore da Figura 2.7, onde "E n" significa "espécie n" e "A n" significa "Ancestral n".

A raiz é um nó especial que não possui pai (Manber, 1989). Por isto a coluna 9 da matriz de adjacências da Figura 2.8 possui apenas zeros. Por exemplo, o algarismo 1 no cruzamento da coluna “E 2” com a linha “A 4” significa que o elemento “E 2” é descendente do elemento “A 4”.

Operador de permutação entre folhas

O operador de permutação entre folhas (ou espécies) trabalha trocando duas colunas posicionadas à esquerda da coluna da raiz. Todas as folhas encontram-se posicionadas nestas colunas iniciais.

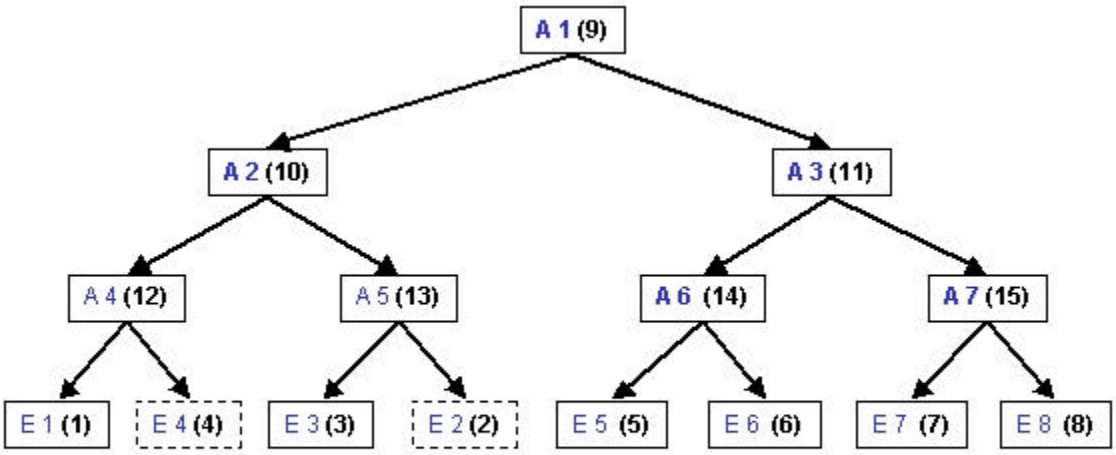


Figura 2.9 Representação gráfica da árvore da Figura 2.7 após aplicação do operador de mutação para troca de colunas à esquerda da raiz, ou seja, troca de espécies. Neste caso foram trocadas as colunas 2 e 4.

Filhos

Pais

	E 1 (1)	E 2 (2)	E 3 (3)	E 4 (4)	E 5 (5)	E 6 (6)	E 7 (7)	E 8 (8)	A 1 (9)	A 2 (10)	A 3 (11)	A 4 (12)	A 5 (13)	A 6 (14)	A 7 (15)
A 1 (9)	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
A 2 (10)	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
A 3 (11)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
A 4 (12)	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
A 5 (13)	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
A 6 (14)	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
A 7 (15)	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

Figura 2.10 Representação na forma de matriz de adjacências da árvore da Figura 2.9, onde "E n" significa "espécie n" e "A n" significa "Ancestral n".

No exemplo das Figuras 2.9 e 2.10, foram trocadas as colunas 2 e 4. Isto significa que após a troca, o elemento “E 2” passou a ser descendente do elemento “A 5” e o elemento “E 4” passou a ser descendente do elemento “A 4”. Antes desta troca, o elemento “E 2” era descendente do elemento “A 4” e o elemento “E 4” era descendente do elemento “A 5”, como pode ser observado na Figura 2.7.

Operador de permutação entre ramos

O operador de permutação entre ramos (ou sub-árvores) trabalha trocando duas colunas posicionadas à direita da coluna da raiz. Todas as sub-árvores encontram-se posicionadas nestas colunas finais.

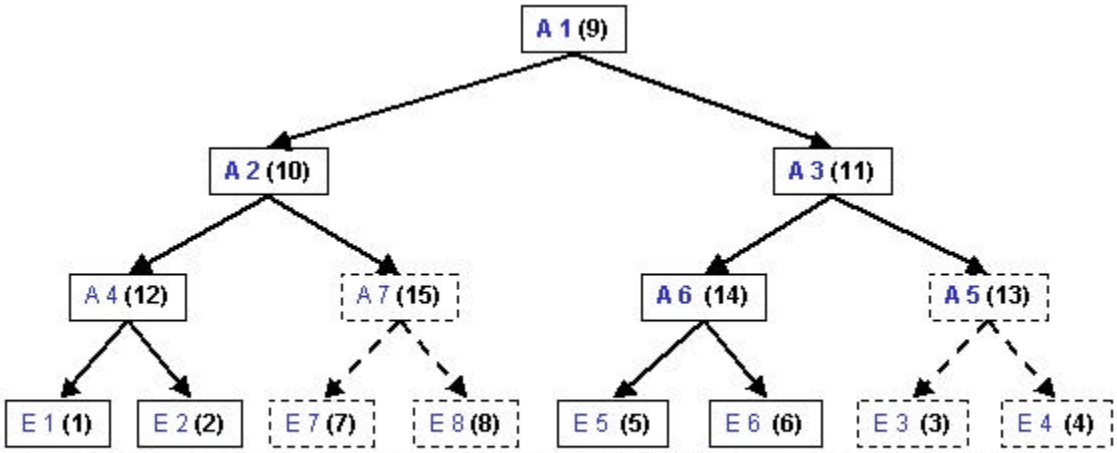


Figura 2.11 Representação gráfica da árvore da Figura 2.7 após aplicação do operador de mutação para troca de colunas à direita da raiz, ou seja, troca de ramos. Neste caso foram trocadas as colunas 13 e 15.

Filhos

Pais

	E1 (1)	E2 (2)	E3 (3)	E4 (4)	E5 (5)	E6 (6)	E7 (7)	E8 (8)	A1 (9)	A2 (10)	A3 (11)	A4 (12)	A5 (13)	A6 (14)	A7 (15)
A1 (9)	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
A2 (10)	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
A3 (11)	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
A4 (12)	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
A5 (13)	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
A6 (14)	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
A7 (15)	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

Figura 2.12 Representação na forma de matriz de adjacências da árvore da Figura 2.11, onde "E n" significa "espécie n" e "A n" significa "Ancestral n".

No exemplo das Figuras 2.11 e 2.12, foram trocadas as colunas 13 e 15. Isto significa que após a troca, a sub-árvore “A 5” passou a ser descendente da sub-árvore “A 3” e a sub-árvore “A 7” passou a ser descendente da sub-árvore “A 2”. Antes desta troca, a sub-árvore “A 5” era descendente da sub-árvore “A 2” e a sub-árvore “A 7” era descendente da sub-árvore “A 3”.

O próximo exemplo mostra que é possível trocar ramos de níveis diferentes da árvore. Isto provoca grande alteração na topologia da árvore. Retomando a árvore da Figura 2.7 e trocando as colunas 10 e 15, teremos a árvore representada na Figura 2.11a.

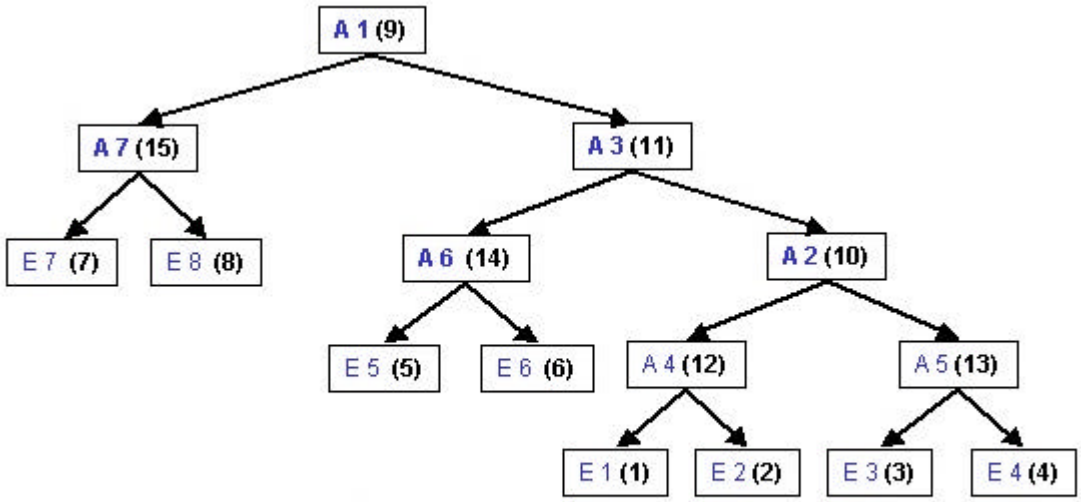


Figura 2.11a Representação gráfica da árvore da Figura 2.7 após aplicação do operador de mutação para troca de colunas à direita da raiz, ou seja, troca de ramos. Neste caso foram trocadas as colunas 10 e 15.

	E1 (1)	E2 (2)	E3 (3)	E4 (4)	E5 (5)	E6 (6)	E7 (7)	E8 (8)	A1 (9)	A2 (10)	A3 (11)	A4 (12)	A5 (13)	A6 (14)	A7 (15)
A 1 (9)	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
A 2 (10)	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
A 3 (11)	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
A 4 (12)	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
A 5 (13)	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
A 6 (14)	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
A 7 (15)	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

Figura 2.12a Representação na forma de matriz de adjacências da árvore da Figura 2.11a, onde "E n" significa "espécie n" e "A n" significa "Ancestral n".

No exemplo das Figuras 2.11a e 2.12a, foram trocadas as colunas 10 e 15. Isto significa que após a troca, a sub-árvore “A 7” passou a ser descendente direta da raiz “A 1” e a sub-árvore “A 2” passou a ser descendente da sub-árvore “A 3”. Antes desta troca, a sub-árvore “A 2” era descendente da raiz “A 1” e a sub-árvore “A 7” era descendente da sub-árvore “A 3”. Este exemplo reforça o poder deste operador para alterar a topologia de uma árvore.

Operador de permutação de ramos com folhas

O operador de permutação entre folhas (ou espécies) e ramos (ou sub-árvores) trabalha trocando duas colunas sendo uma posicionada à esquerda da coluna da raiz e a outra posicionada à direita da coluna da raiz.

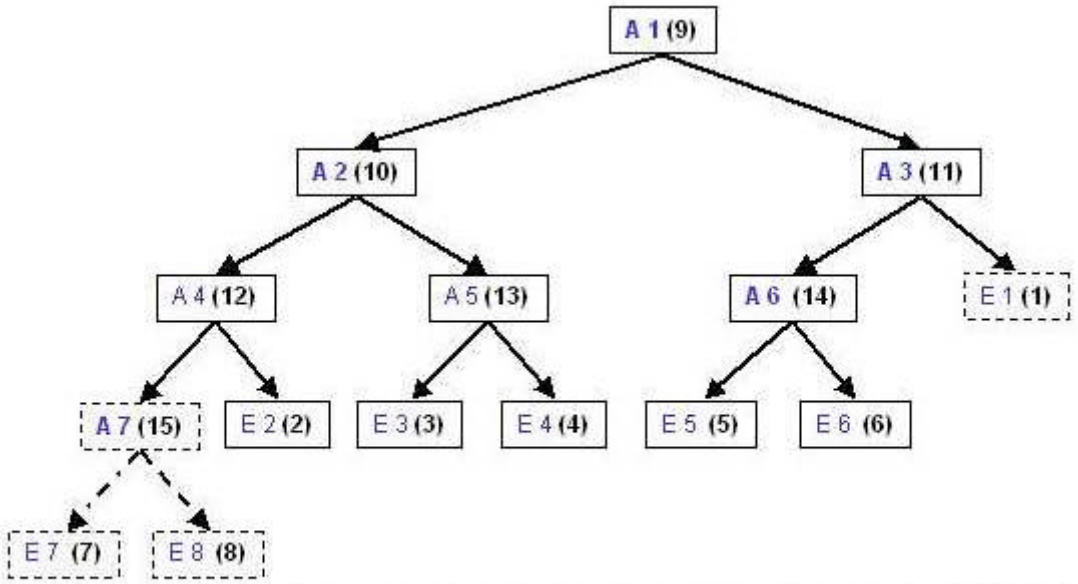


Figura 2.13 Representação gráfica da árvore da Figura 2.7 após aplicação do operador de mutação para troca de uma coluna à direita da raiz por outra à sua esquerda, ou seja, troca de ramo por folha. Neste caso foram trocadas as colunas 1 e 15.

Filhos

Pais

	E 1 (1)	E 2 (2)	E 3 (3)	E 4 (4)	E 5 (5)	E 6 (6)	E 7 (7)	E 8 (8)	A 1 (9)	A 2 (10)	A 3 (11)	A 4 (12)	A 5 (13)	A 6 (14)	A 7 (15)
A 1 (9)	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
A 2 (10)	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
A 3 (11)	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0
A 4 (12)	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
A 5 (13)	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
A 6 (14)	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
A 7 (15)	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

Figura 2.14 Representação na forma de matriz de adjacências da árvore da Figura 2.11, onde "E n" significa "espécie n" e "A n" significa "Ancestral n".

No exemplo das Figuras 2.11 e 2.12, foram trocadas as colunas 1 e 15. Isto significa que após a troca, o elemento “E 1” passou a ser descendente da sub-árvore “A 3” e a sub-árvore “A 7” passou a ser descendente da sub-árvore “A 4”. Antes desta troca, o elemento “E 1” era descendente da sub-árvore “A 4” e a sub-árvore “A 7” era descendente da sub-árvore “A 3”.

2.8 Algoritmos meméticos

Introdução

Os cinco anos em que Charles Darwin atuou como biólogo em sua viagem a bordo do navio Beagle, permitiram que ele coletasse uma grande quantidade de informação e material biológico. Toda essa massa de dados, em conjunto com as suas observações, certamente contribuíram para a formulação das suas idéias sobre os mecanismos da evolução (Silva & Sasson, 1996). Pode-se inferir que Darwin, ao elaborar a sua Teoria da Evolução, também expôs suas idéias (um conjunto de observações, conhecimento adquirido, genialidade e conclusões) a um processo de seleção. Ele construiu uma "pré-história" sempre confrontando as conclusões obtidas com os fatos observados e com os dados coletados. Este método de trabalho fez com que as características em comum fossem mantidas, e as outras, descartadas. Na verdade, Darwin trabalhava com uma população de idéias, ou em outras palavras, "indivíduos de concepção mental" que passavam por um processo de busca pelo melhor entendimento dos dados coletados. É também provável que, com o conhecimento adquirido ao longo do tempo, ele tenha aperfeiçoado seus mecanismos de busca e seus critérios de avaliação. Este processo de busca pela melhor resposta somente pode ser concebido dentro dos limites do campo do conhecimento, na forma de um processo iterativo aplicado por um indivíduo para a melhoria continuada de suas idéias. Este processo sofisticado de refinamento do conhecimento se justifica no caso de problemas que envolvem um grande número de possibilidades de interpretação e formalização.

Memes

No campo da computação evolutiva, o refinamento do conhecimento pode ser incorporado como uma etapa de apoio ao processo evolutivo. Moscato & Norman (1992) introduziram o termo "algoritmo memético" para descrever um processo evolutivo que possua uma busca local como parte decisiva na evolução. Essa busca pode ser caracterizada como sendo um refinamento local dentro de um espaço de busca, de modo que um indivíduo pode ter seu nível de adaptação aumentado após este refinamento. O termo meme foi idealizado por Dawkins (1976) como sendo uma unidade de informação que se reproduz durante um processo argumentativo e de transmissão de conhecimento (Radcliffe & Surry, 1994). Portanto, pode-se dizer que, enquanto o algoritmo memético está relacionado com a evolução cultural, o algoritmo genético está baseado na evolução biológica dos indivíduos.

Uma diferença marcante entre genes e memes está no processo de transmissão aos seus descendentes. Quando o meme é transmitido, ele será adaptado pela entidade que o recebe com base no seu conhecimento e para melhor atender a suas necessidades. Quanto aos genes, no processo de evolução eles são transmitidos de uma maneira tal que o descendente gerado vai herdar muitas habilidades e características presentes em seus progenitores. Dessa maneira, o algoritmo genético é inspirado na tentativa de emulação computacional da evolução biológica e o algoritmo memético tenta fazer o mesmo em relação à evolução cultural.

É interessante lembrar que na evolução biológica a informação está codificada nos genes por uma seqüência de nucleotídeos, de modo que a transmissão é influenciada pela presença de mutação, recombinação e pela seleção natural em relação aos indivíduos geneticamente melhor adaptados. Na evolução cultural, a informação envolvida está nos memes, de modo que as alterações surgem pela combinação, criação e reorganização das representações mentais (conscientes ou não) e pela possível ineficácia dos mecanismos de transmissão de informação. A replicação (fenótipo) ocorre quando essas representações mentais são transformadas em ações passíveis de imitação ou expressas através de alguma linguagem. A incorporação dessa nova informação por parte de algum indivíduo certamente alterará a pressão seletiva e a influência vinculada às limitações impostas pelo ambiente àquele indivíduo.

Aspectos de implementação de algoritmos meméticos

De acordo com Moscato (1999), o uso genérico da denominação de Algoritmo Memético é feito para a identificação de uma classe de meta-heurísticas, constituindo um dos procedimentos de maior sucesso para problemas de otimização combinatória.

A característica principal, presente em muitas implementações que utilizam algoritmos meméticos, é o uso de processos de busca dedicados. Esses processos pretendem utilizar toda a informação disponível sobre o problema, de modo que este conhecimento seja incorporado sob a forma de heurísticas, técnicas de busca local, operadores especializados de recombinação e muitas outras maneiras.

Em essência, os algoritmos meméticos podem ser interpretados como um conjunto de estratégias que implementam a competição e a cooperação entre diferentes mecanismos de otimização (Moscato & Norman, 1992). Sendo assim, o sucesso obtido pode ser explicado como sendo uma conseqüência direta da sinergia dos diferentes processos de busca utilizados.

A idéia geral dos algoritmos meméticos é a utilização dos operadores evolutivos que determinam regiões promissoras no espaço de busca, combinados com busca local nestas

regiões (este processo tem sido aplicado com sucesso em vários problemas de otimização – Merz & Freisleben, 1999). Também se pode dizer que algoritmos meméticos correspondem à união de um método de busca global e uma heurística local aplicada a cada indivíduo, de modo que um algoritmo memético é, na verdade, um tipo especial de *hillclimbing*.

Conforme Merz & Freisleben (1999), em um ambiente que emprega algoritmo memético, os operadores de recombinação e mutação agem como estratégias de diversificação. Os indivíduos da população podem estar localizados em uma região do espaço de busca contendo um ótimo local, chamada base de atração do ótimo local. Utilizando a informação contida na população, novos pontos de partida podem ser descobertos após a busca local. Os operadores de recombinação e mutação podem gerar indivíduos da população que estejam localizados em bases de atração de ótimos locais ainda não explorados, de modo que um novo pico deva ser alcançado (maximização) ou um vale deva ser explorado (minimização). Utilizando o conceito de superfície de adaptação (*fitness*), a Figura 2.15 ilustra estes eventos, no caso da maximização. Após a recombinação, o filho gerado pode possuir um *fitness* baixo, mas um grande potencial para crescimento, de modo que uma busca local pode levar o descendente a assumir um valor de *fitness* elevado. A mutação, por sua vez, pode levar a um pequeno aumento (ou decréscimo) do *fitness*, por representar uma perturbação local junto à representação do indivíduo.

Figura 2.15 Operadores de recombinação e mutação agindo como estratégias de diversificação junto a algoritmos meméticos.

Na presença de restrições, os operadores genéticos de recombinação e mutação podem produzir soluções que estão fora da região factível do espaço de busca. Entretanto, um algoritmo de reparação (factibilização) pode ser elaborado para remeter o descendente gerado para uma região factível (Radcliffe & Surry, 1994). A razão para a distinção entre uma busca local realizada por um algoritmo memético e a busca realizada por um algoritmo de reparação é

que este último não investe no aumento da qualidade da solução, mas apenas na garantia de sua factibilidade.

No caso de problemas de otimização irrestritos, a geração da população inicial é realizada em duas etapas. A primeira, consiste na geração de indivíduos que procuram explorar amplamente o espaço de busca, caracterizando-os como soluções candidatas. Na segunda etapa, procedimentos de busca local são implementados para a obtenção de um ótimo associado à região do espaço em que se encontra cada indivíduo. Esse mesmo processo de busca é empregado após a aplicação de qualquer operador genético (Merz & Freisleben, 1999), procurando remeter o descendente gerado a um ótimo local.

No caso de problemas de otimização com restrições, já na geração da população inicial podem surgir indivíduos infactíveis, não porque possuam um genótipo mal formado ou porque o algoritmo de solução utilizado contenha erros, mas sim por violarem restrições impostas ao problema. Há, portanto, a necessidade de algum dispositivo de seleção que seja capaz de discernir entre indivíduos factíveis ou não. Em algumas aplicações, é possível implementar algoritmos de reparação para levar indivíduos à factibilidade (eventualmente à região factível mais próxima). Uma vez tendo essa necessidade atendida, é uma conseqüência natural o emprego de uma busca local sobre os indivíduos factíveis, fazendo com que eles possam atingir um ótimo local dentro da região factível. Portanto, resulta como abordagem de solução um algoritmo memético (ou algoritmo genético associado a mecanismos de busca local) aliado a etapas de reparação, tanto para conduzir os indivíduos à factibilidade como para aumentar o seu *fitness*.

Formalização do Processo de Busca Local

Aarts & Verhoeven (1997) consideram a busca local como sendo uma aproximação geral para problemas de otimização combinatória baseados na exploração de vizinhanças (para maiores detalhes veja Aarts & Lenstra, 1997).

De maneira genérica, um algoritmo de busca local começa com um indivíduo $s_0 \in S$ e tenta continuamente encontrar melhores elementos dentre os vizinhos. Em outras palavras, a finalidade será remeter esse indivíduo para um local onde a função de *fitness* tenha um valor melhor que o atual (Figura 2.15). Essa busca deverá ser realizada até que uma condição de parada seja satisfeita, sendo que essa condição deve ser atendida sempre que o processo de

busca não tenha mais a capacidade de melhorar a solução atual. Observe o algoritmo representativo de um processo de busca na Figura 2.16

Procedimento busca local

Início

Repita

Final ← falso

Para i ← 1 **até** número de indivíduos **faça**

Início

Elemento ← indivíduo(i)

Aplicar busca local(elemento)

Se *fitness* é pior **então** Final ← verdadeiro

Fim

Até Final = falso

Figura 2.16 Algoritmo genérico representativo de uma busca local.

Moscato (1999) apresenta uma definição formal de busca local que será apresentada a seguir.

Um problema computacional P tem domínio de entrada I_p, com $x \in I_p$, podendo ser estabelecido um conjunto $ans_p(x)$ de respostas correspondentes. Entretanto, é preciso garantir que exista um subconjunto $sol_p(x) \subseteq ans_p(x)$ que identifica as soluções factíveis de P. Um algoritmo soluciona um problema P se, para a entrada $x \in I_p$, apresentar como saída $y \in sol_p(x)$ – solução factível – ou, no caso de $sol_p(x) = \{\}$, indicar que não existe y. A otimização combinatória é um tipo especial de problema de busca, em que cada $x \in I_p$ tem um conjunto $sol_p(x)$ de cardinalidade finita e cada solução $y \in sol_p(x)$ tem um valor de *fitness* $m_p(y,x)$. A busca, nesse tipo de problema, será responsável por encontrar uma solução factível $y^* \in sol_p(x)$ que maximize o *fitness* $m_p(y,x)$.

Uso de meméticos para busca de árvores

A ferramenta computacional desenvolvida para esta pesquisa de mestrado (Projeto Árvores Filogenéticas) explora técnicas apresentadas nesta seção quando está preparando a população intermediária. À medida que novos elementos são gerados, é realizada busca local tentando melhorar seu *fitness,* e somente se o novo elemento apresentar alguma melhora em relação a seus ancestrais, ele é passado para a próxima geração. Para que não haja problemas de terminação junto aos procedimentos de busca local, causado por exemplo pela presença de mínimos locais, existe um limite de tentativas por elemento, que pode ser configurado pelo usuário na tela de opções da ferramenta. O número padrão é dez tentativas e vem apresentando bons resultados até o momento.

A implementação destas técnicas no Projeto Árvores Filogenéticas foi fundamental para acelerar o tempo de convergência do algoritmo genético. Quando o parâmetro que limita as tentativas para melhorar cada novo elemento é deixado em zero, o tempo de convergência e o número de gerações necessário para encontrar uma boa árvore sobem consideravelmente.

Darwinismo x Lamarkismo

Após a busca local tentando melhorar cada novo elemento, não é realizada nenhuma operação diretamente sobre seu genótipo, por isto, a abordagem adotada pela ferramenta foi o darwinismo.

Teria sido o lamarkismo se, após a busca local, fosse alterado o genótipo de algum elemento.

Conclusões

Os três operadores de mutação propostos (permutação de ramos com ramos, de ramos com folhas e de folhas com folhas) são capazes promover qualquer tipo de alteração junto à estrutura de uma árvore, sem oferecer grandes riscos à sua integridade. Com seu apoio, é possível explorar adequadamente o espaço de solução do problema das árvores filogenéticas, em busca de uma boa solução para o conjunto de seqüências em análise. O uso destes operadores genéticos, associado a procedimentos de busca local visando melhorar os resultados parciais, forneceu grande poder de busca e bom desempenho ao Projeto Árvores Filogenéticas. Os conceitos apresentados neste capítulo foram utilizados, direta ou indiretamente, no projeto de software e na análise dos resultados obtidos.

Capítulo 3

Filogenia

3.1 Introdução

Segundo a Enciclopédia Mirador Internacional (1995), a palavra Filogenia foi criada pelo naturalista alemão Ernst Heinrich Haeckel (1834 – 1919).

Para alguns autores, filogenia é o estudo da evolução dos grupos dos seres vivos; para outros, é a história evolucionária desses grupos; outros por fim preferem defini-la como história da vida (Enciclopédia Mirador Internacional, 1995).

A idéia de estabelecer classificações filogenéticas nasceu realmente com Darwin. Ninguém mais empenhado do que ele em revelar a verdade da origem das espécies e a eficácia da seleção natural, como seu principal mecanismo. Mas, embora tenha sugerido ramificações que poderiam ter ocorrido em diversos grupos, não cuidou de organizar num todo coerente as possíveis alterações evolutivas e assim aplicar a filogenia a um sistema geral de classificação. Não lhe interessava muito, aliás, o trabalho de classificar; preocupava-se mais em analisar classificações já existentes, realizadas antes da sua teoria, para mostrar que a diversificação dos grupos bem poderia ter sido originada por evolução (Enciclopédia Mirador Internacional, 1995).

3.2 Representação da filogenia

O primeiro diagrama foi possivelmente o de Lamark, em 1809. Um diagrama desse gênero é o único desenho que se encontra na *Origem das Espécies* (1859) de Darwin. Em 1866, E. Haeckel fez o que Darwin não realizara: uma tentativa de filogenia geral, abrangendo todos os grupos de seres vivos – microorganismos, plantas e animais; representou-a por uma árvore, que foi concebida com base em aspectos muito questionáveis do ponto de vista científico e que não pode comparar-se tecnicamente às que modernamente se organizam (Enciclopédia Mirador Internacional, 1995).

Para Reijmers *et al.* (1999), filogenia pode ser melhor explicada como sendo uma estrutura em forma de árvore que define certos relacionamentos ancestrais entre conjuntos relacionados de objetos (proteínas, espécies, etc.). O nó na base da árvore, denominado raiz, representa o ancestral em comum a todos os objetos relacionados.

3.3 Árvore Gênica e Árvore de Espécies

Segundo Nei & Kumar (2000), os evolucionistas estão geralmente interessados em uma árvore filogenética que represente a história evolucionária de um grupo de espécies ou populações. Este tipo de árvore é chamado árvore de espécies ou árvore de populações. Entretanto, quando uma árvore filogenética é construída a partir de genes de cada espécie, a árvore obtida não necessariamente coincide com a árvore de espécies. Para distinguir esta árvore da anterior, ela é chamada árvore gênica.

A primeira distinção entre estes dois tipos de árvores foi feita por Tateno *et al.* (1982). Na Figura 3.1, a espécie B é ancestral de X e Y e a espécie A é ancestral de B e Z (Weir, 1996).

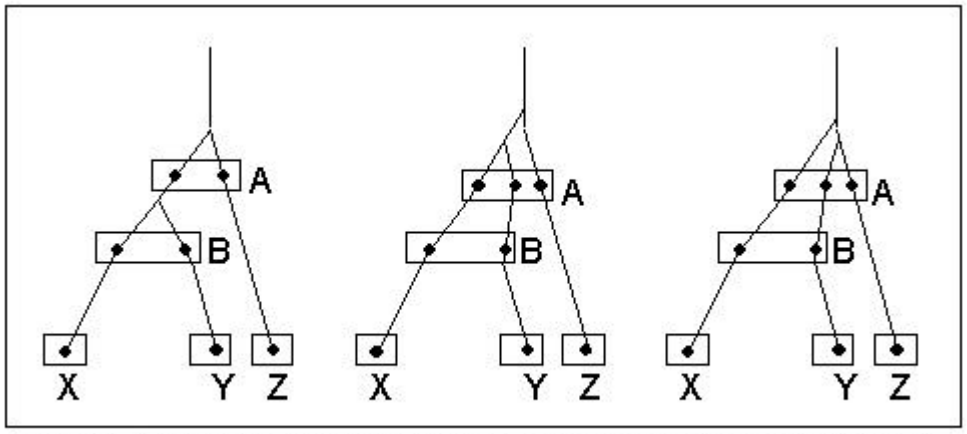


Figura 3.1 Árvores gênicas citadas por Nei (1987), Nei & Kumar (2000) e Weir (1996)

A árvore de espécies une as espécies A, B, X, Y e Z. Três possíveis árvores gênicas, indicadas pelas linhas sólidas, ligam os genes indicados pelos círculos sólidos (Nei, 1987; Nei & Kumar, 2000; Weir, 1996).

As árvores gênicas da esquerda e do centro possuem genes nas espécies X e Y mais proximamente relacionados uns com os outros do que os genes na espécie Z enquanto que na

árvore gênica da direita os genes das espécies Y e Z estão mais proximamente relacionados (Weir, 1996).

3.4 Árvore com raiz e sem raiz

No apêndice B são comentados brevemente os cálculos que conduzem aos números de árvores, de ramos e de nós a partir da quantidade de folhas em análise.

Na Figura 3.2, podem ser observadas duas das 15 possíveis árvores com raiz e que apresentam quatro folhas. Elas contêm a noção de ordem temporal, enquanto que a única árvore sem raiz dentre as três possíveis, apresenta apenas a noção de distância sem noção alguma sobre ancestrais ou descendentes (Weir, 1996).

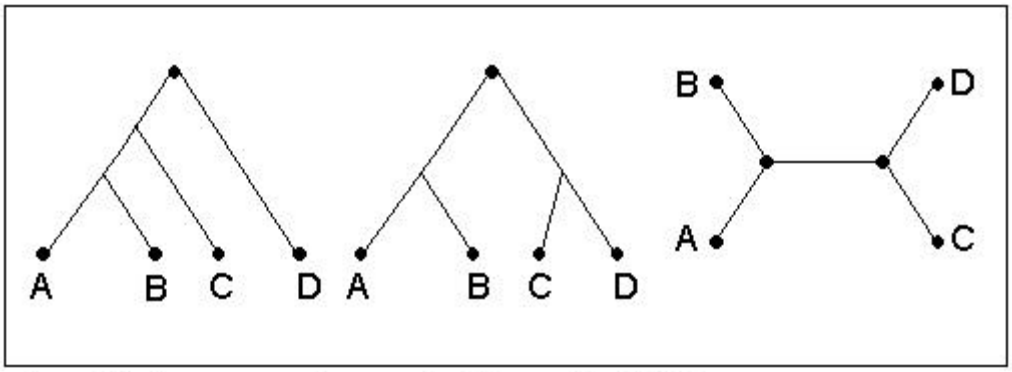


Figura 3.2 Árvores com raiz e sem raiz citadas por Weir (1996)

Ao longo deste trabalho sempre serão consideradas árvores com raiz.

3.5 Métodos baseados e não baseados em modelo

A distinção entre os tipos de métodos utilizados para busca das árvores filogenéticas foi feita por Swofford *et al*. (1996). Ele estabeleceu que, para os métodos não baseados em modelo, deve haver um algoritmo que conduza até a árvore desejada. Para os métodos baseados em modelo deve existir um critério pelo qual as árvores candidatas serão analisadas e avaliadas até que seja encontrada a de maior verossimilhança.

Uma característica dos métodos não baseados em modelo é que eles determinam apenas uma árvore como solução do problema em análise, e não fornecem informações sobre outras

árvores candidatas que poderiam representar outras boas soluções para o problema. Os métodos baseados em modelo geralmente examinam grande quantidade de árvores candidatas durante o processo de busca daquela que maximiza o critério (Weir, 1996).

3.6 Métodos não baseados em modelo

São métodos baseados apenas em critérios de otimização específicos. Eles não requerem modelos explícitos para sua formulação (Swofford *et al*., 1996).

3.6.1 Matriz de Distâncias

Os métodos que empregam matriz de distâncias são baseados nos conjuntos de distâncias evolucionárias calculadas entre cada par de espécies. A árvore filogenética é construída levando em consideração os relacionamentos entre estas distâncias. Estas distâncias geralmente se referem ao número de alterações entre as espécies. A qualidade das árvores resultantes depende da qualidade das distâncias estimadas (Nei & Kumar, 2000; Weir, 1996).

O processo de busca da melhor árvore filogenética também pode ser entendido como um processo de agrupamento. Agrupamento pode ser definido, a grosso modo, como sendo o processo de organizar objetos em grupos cujos membros são similares de alguma forma. Para maiores informações especificamente sobre Clusterização, veja os trabalhos de Everitt (1993), Rasmussen (1992), Kaufman & Rousseeuw (1990), Jain & Dudes (1988) e Gordon (1981).

3.6.1.1 Método UPGMA

O método mais simples deste grupo (baseado em matriz de distâncias) é conhecido como *unweighted pair-group method using an arithmetic average*. Conforme Nei & Kumar (2000), este método é atribuído a Sneath & Sokal (1973).

A árvore construída baseada neste método também é conhecida como ***fenograma***, porque ela foi originalmente utilizada para representar a extensão das similaridades fenotípicas de um grupo de espécies em taxonomia numérica. Entretanto, ela também pode ser usada para construir filogenias moleculares quando a taxa de substituição de gene for mais ou menos constante. Particularmente quando dados de freqüência genética são usados para reconstrução

filogenética, este modelo produz bons e confiáveis resultados comparados com outros métodos baseados em matriz de distâncias (Nei *et al.*, 1983; Takezaki & Nei, 1996).

O UPGMA se propõe a construir árvores de espécies, embora decisões incorretas na construção da árvore freqüentemente ocorram quando a taxa de substituição de gene não é constante ou quando o número de genes ou nucleotídeos é pequeno (Nei & Kumar, 2000).

Algoritmo

Segundo Nei & Kumar (2000), neste método uma medida da distância evolucionária é computada para todos os pares de seqüências, e os valores destas distâncias são apresentados em formato matricial, como mostrado a seguir, onde d_{ij} representa a distância entre o i-*ésimo* e o j-*ésimo* elementos.

$$\begin{bmatrix} & 1 & 2 & 3 & 4 \\ 2 & d_{12} & & & \\ 3 & d_{13} & d_{23} & & \\ 4 & d_{14} & d_{24} & d_{34} & \\ 5 & d_{15} & d_{25} & d_{35} & d_{45} \end{bmatrix}$$

O processo de clusterização começa com o par de elementos que apresentar a menor distância. Supondo que d_{12} seja a menor distância encontrada na matriz anterior, os elementos 1 e 2 serão agrupados com um ramo de comprimento $b_1 = \frac{d_{12}}{2}$. Os elementos 1 e 2 são agora combinados em um único agrupamento [u = (1-2)], e a distância entre u e outros elementos é calculada por $d_{uk} = \frac{d_{1k} + d_{2k}}{2}$. Desta operação resulta a matriz a seguir.

$$\begin{bmatrix} & u = (1-2) & 3 & 4 \\ 3 & d_{u3} & & \\ 4 & d_{u4} & d_{34} & \\ 5 & d_{u5} & d_{35} & d_{45} \end{bmatrix}$$

Agora supondo que a distância d_{u3} seja a menor da matriz, os elementos u e 3 são combinados em um novo agrupamento [v = (1-2-3)], com um ramo de comprimento

$b_2 = \frac{d_{u3}}{2} = \frac{d_{13} + d_{23}}{2 * 2}$. A distância entre v e outros elementos é calculada por $d_{uk} = \frac{d_{k1} + d_{k2} + d_{k3}}{3}$. A nova matriz é apresentada a seguir.

$$\begin{bmatrix} & v = (1-2-3) & 4 \\ 4 & d_{v4} & \\ 5 & d_{v5} & d_{45} \end{bmatrix}$$

Supondo, a partir deste ponto, que a menor distância da matriz acima seja d_{v4}, é necessário combinar v com 4, com ramo de comprimento $b_3 = \frac{d_{v4}}{2} = \frac{d_{14} + d_{24} + d_{34}}{3 * 2}$. Como só restou um elemento, ele deve ser agrupado com os demais, e o comprimento de seu ramo será dado por $b_4 = \frac{d_{15} + d_{25} + d_{35} + d_{45}}{4 * 2}$. Na Figura 3.3, pode ser visualizada uma representação gráfica para estes agrupamentos.

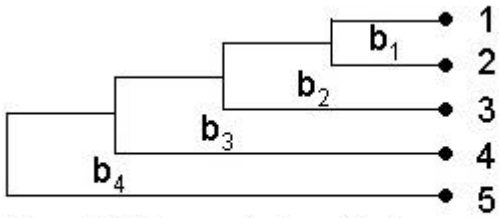


Figura 3.3 Representação gráfica do agrupamento obtido com o método UPGMA (Nei & Kumar, 2000)

Em Nei & Kumar (2000), pode-se verificar que a fórmula para se determinar a distância entre dois agrupamento A e B é dada por:

$$d_{AB} = \sum_{ij} \frac{d_{ij}}{rs}$$

onde r e s são os números de elementos nos agrupamentos A e B, respectivamente, e d_{ij} é a distância entre o elemento i no agrupamento A e o elemento j no agrupamento B.

3.6.1.2 Quadrados mínimos

Quando a taxa de substituição de nucleotídeos varia de uma linhagem evolucionária para outra, UPGMA geralmente encontra topologias incorretas. Nestes casos, devem ser usados

métodos que permitem taxas diferentes de variações de substituição de nucleotídeos para ramos diferentes. Uma alternativa é usar o método dos quadrados mínimos. Existem vários métodos baseados em quadrados mínimos, mas os mais comuns são o ordinário e o ponderado (Nei & Kumar, 2000).

Ordinário

No método dos quadrados mínimos ordinário para inferência filogenética (Cavalli-Sforza & Edwards, 1967), deve-se analisar a seguinte soma residual dos quadrados:

$$R_s = \sum_{i<j} (d_{ij} - e_{ij})^2$$

onde d_{ij} e e_{ij} são as distâncias observadas e esperadas entre os elementos *i* e *j*, respectivamente. A distância esperada entre os elementos *i* e *j* é a soma das estimativas dos comprimentos de todos os ramos conectando os dois elementos em uma árvore. Por exemplo, a distância esperada entre homens e gorilas na árvore da Figura 3.4 é $a + c + d = 0{,}110$.

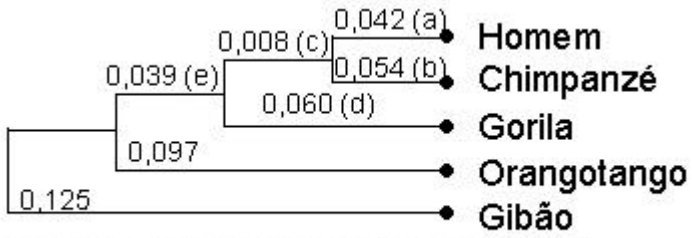


Figura 3.4 Distâncias de Kimura (Nei & Kumar, 2000)

No método dos quadrados mínimos ordinários, R_s deve ser calculado para todas as possíveis topologias e a topologia que apresentar o menor valor para R_s deve ser escolhida como sendo a árvore final (Nei & Kumar, 2000).

Ponderado (Fitch-Margoliash)

Fitch & Margoliash (1967) usaram a seguinte fórmula para R_s:

$$R_s = \sum_{i<j} \frac{(d_{ij} - e_{ij})^2}{d_{ij}}$$

onde $\frac{1}{d_{ij}}$ é o termo de ponderação. Com isso, uma medida de erro absoluto passa a ganhar características de uma medida de erro relativo.

Este método também é conhecido como quadrados mínimos ponderados. Na prática, os valores de R_s resultantes das duas fórmulas anteriores podem conduzir a topologias similares (Nei & Kumar, 2000).

3.6.1.3 Evolução Mínima

Segundo Nei & Kumar (2000), para este método devem ser calculadas as somas (S) de todas as estimativas de comprimentos de todos os ramos para cada uma das árvores candidatas. A árvore que apresentar o menor valor de S será a escolhida.

$$S = \sum_{i}^{T} \hat{b}_i \text{, onde:}$$

$\hat{b}_i$ representa a estimativa de comprimento do ramo *i* e;

T é o total de ramos, ou seja, 2m-3;

"m" é o número de folhas da árvore.

Este método foi inicialmente proposto por Edwards & Cavalli-Sforza (1963), mas sua fundamentação teórica foi feita por Rzhetsky & Nei (1993), onde eles demonstraram que o valor esperado de S se torna menor para a verdadeira topologia, independente do número de seqüências estudadas (Nei & Kumar, 2000).

Entretanto, como este método necessita examinar todas as árvores candidatas, ele é excessivamente dispendioso em termos de tempo computacional, e foi por esta razão que Rzhetsky & Nei (1993) sugeriram que primeiro fosse construído um menor número de árvores candidatas baseadas no método *Neighbor-Joining,* para depois submeter as selecionadas ao método da evolução mínima (Nei & Kumar, 2000).

3.6.1.4 Neighbor-Joining

Saitou & Nei (1987) desenvolveram um eficiente método de construção de árvores filogenéticas baseado no princípio da evolução mínima. Este método não examina todas as possíveis topologias, mas em cada estágio de ramificação da árvore o princípio da evolução

mínima é utilizado. Este método é considerado uma simplificação do método da evolução mínima, descrito anteriormente neste capítulo (Nei & Kumar, 2000).

Saitou & Nei (1987) descreveram um método para identificar os pares mais próximos de elementos ou vizinhos (*neighbors*), de forma a minimizar o comprimento total da árvore construída. Um par de vizinhos é definido como sendo formado por dois elementos conectados por um ramo em uma árvore sem raiz bifurcada (dois ramos unidos por um nó interior). Na Figura 3.5, homem e chimpanzé são vizinhos, mas homem e gorila não são. Se homem e chimpanzé forem combinados em um único nó, então esta combinação e gorila se tornam vizinhos. Em geral é possível determinar a topologia de uma árvore por uniões sucessivas de pares de vizinhos (Weir, 1996).

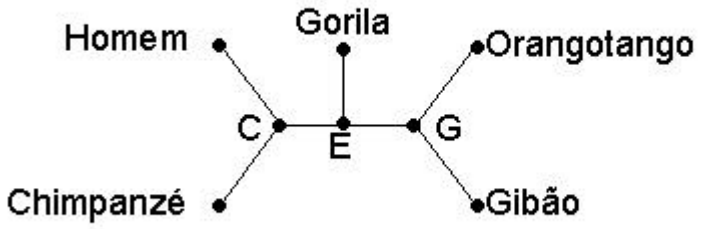


Figura 3.5 Árvore sem raiz obtida com o método *Neighbor-joining* (Weir, 1996)

Este método começa com uma estrutura em forma de estrela, como a mostrada na Figura 3.6. Os vizinhos são os pares de espécies que, quando combinados, resultam em uma árvore de menor comprimento total. Eles devem ser unidos a fim de formar uma nova unidade (composta pelos dois vizinhos identificados). Este processo de identificação de vizinhos é repetido até que existam apenas três elementos (combinados) na estrutura (Weir, 1996).

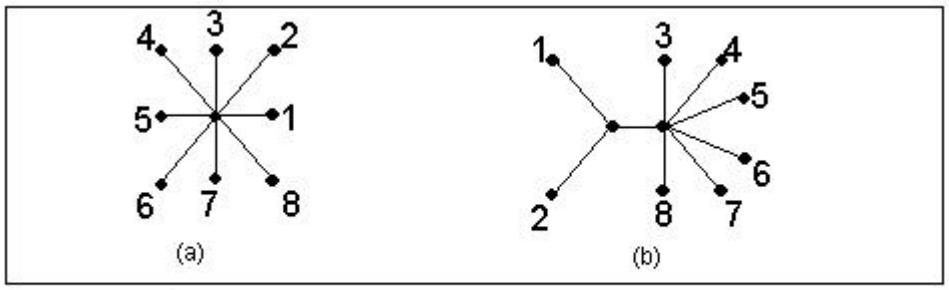


Figura 3.6 (a) Árvore sem raiz em forma de estrela sem estrutura hierárquica; (b) árvore sem raiz em que os elementos 1 e 2 foram agrupados (Weir, 1996; Saitou & Nei, 1987)

Saitou & Nei (1987) mostraram que este método produz a uma boa árvore por pura adição de dados, onde a distância entre cada par de espécies é a soma dos comprimentos dos ramos que os unem na árvore (Weir, 1996).

Kumar (1996), desenvolveu um algoritmo baseado neste método que é capaz de gerar mais de uma topologia, levando em consideração não apenas os melhores candidatos a vizinhos mas também alguns outros próximos do mínimo. Com esta alteração, é possível gerar tantas topologias quanto se queira e a melhor delas pode ser escolhida ao final do processo levando em consideração que este método é baseado na análise das distâncias entre seqüências sendo que os pares que apresentarem a menor distância entre si será considerado como sendo um par de vizinhos e serão agrupados por um ancestral em comum. Este é um método híbrido que combina os métodos da Evolução Mínima e *Neighbor-Joining* (Nei & Kumar, 2000).

Conforme demonstrado por Rzhetsky & Nei (1993), é esperado que este método forneça a topologia correta se o número de atributos examinados junto a cada elemento que compõe as folhas da árvore for suficientemente elevado (Nei & Kumar, 2000).

3.6.2 Máxima Parcimônia

Dentre os métodos numéricos existentes, usados para inferir topologias diretamente a partir da matriz de disâncias, os mais utilizados são baseados no princípio da Máxima Parcimônia (Swofford *et al*., 1996).

Segundo Nei & Kumar (2000), Eck & Dayhoff (1966) parecem ter sido os primeiros a utilizar este método para construção de árvores filogenéticas a partir de seqüências de aminoácidos.

A noção de parcimônia na Ciência essencialmente recomenda conservar a hipótese mais simples em detrimento das mais complexas e que as hipóteses *ad hoc* devem ser evitadas sempre que possível. Os métodos baseados em parcimônia assumem que atributos compartilhados por alguns elementos são aqueles herdados de um ancestral em comum. Entretanto, quando ocorrem conflitos junto às características dos atributos não é possível evitar as hipóteses *ad hoc* (Swofford *et al*., 1996). Este método é baseado na filosofia de William de Ockham, que afirma que a melhor hipótese para explicar um processo é aquela que requer a menor quantidade de assunções (Ockham, 1964).

Estes métodos não requerem modelos explícitos de alterações evolucionárias e foram introduzidos por Edwards & Cavalli-Sforza (1963), para estudo de freqüência genética (Weir, 1996).

Para cada topologia possível, o número de seqüências em cada nó é inferido de modo a requerer o menor número de alterações para gerar as seqüências de cada um dos dois nós descendentes. O número total de alterações necessárias para atravessar a árvore toda é então encontrado, e a árvore que leva a um menor número de alterações é escolhida como sendo a de máxima parcimônia. Na ocorrência de múltiplas soluções, outros critérios de desempate devem ser empregados (Swofford *et al*., 1996). Para ilustrar este procedimento, considere o exemplo da Figura 3.7, dado por Fitch (1971).

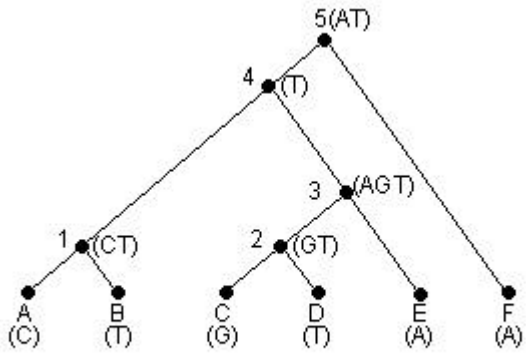


Figura 3.7 Ilustração do processo usado para encontrar a árvore de máxima parcimônia para um sítio (Fitch, 1971; Weir, 1996)

Seqüências de seis espécies , A até F, estão disponíveis para análise. Cada nó de 1 a 5 deve ser analisado, um de cada vez, começando com aqueles que ocupam posições mais próximas das seqüências ainda existentes. Os nós internos (ancestrais) são apenas hipotéticos, apenas os nós externos (descendentes) estão disponíveis para análise. Em cada nó, a "parcimônia" das duas seqüências dos descendentes deve ser anotada. Esta operação, representada por $\Diamond$, é uma operação de conjuntos definida como sendo a interseção de dois conjuntos, desde que não resulte um conjunto vazio, ou a união de dois conjuntos, caso sua interseção resulte em um conjunto vazio. Para ilustrar, considere o ancestral (hipotético) 1 relativo aos descentes A e B da Figura 3.7. A sua parcimônia deverá ser o conjunto (C,T), ou seja, a união das bases de seus descendentes. Considerando agora o ancestral (hipotético) 4, podemos ver que sua parcimônia é (T), ou seja, a intersecção das bases de seus descendentes (Weir, 1996).

Este processo deve ser repetido desde as folhas (descendentes atuais) até a raiz, passando por todos os nós interiores (ancestrais hipotéticos) registrando sua parcimônia. Em seguida caminha-se no sentido contrário, descendo da raiz até as folhas a fim de verificar qual o caminho mais curto para percorrer toda a árvore, ou seja qual a menor quantidade de alterações necessárias para gerar toda a árvore. A árvore de menor comprimento total será considerada a mais parcimoniosa. Caso existam várias alternativas, deve-se adotar algum critério de desempate (Weir, 1996). Este processo deve ser repetido para todos os sítios das seqüências analisadas, pois a árvore escolhida deve ser a mais parcimoniosa para as seqüências inteiras e não somente para um sítio em particular (Weir, 1996).

Para efeito de comparação, são apresentadas a seguir algumas operações tradicionais de conjuntos e a operação de "parcimônia" (Weir, 1996).

Interseção:	$[X,Y] \cap [X,Z] = [X]$	$[X] \cap [Y] = \phi$
União:	$[X,Y] \cup [X,Z] = [X,Y,Z]$	$[X] \cup [Y] = [X,Y]$
Parcimônia:	$[X,Y] \Diamond [X,Z] = [X]$	$[X] \Diamond [Y] = [X,Y]$

O menor número de alterações na árvore montada é o número de uniões nos seus nós internos. O menor número de alterações para esta árvore será quatro se todos os nós internos assumirem o valor T. Este processo deve ser repetido para outras topologias e para outros sítios, e a topologia que requerer o menor número total de alterações, após analisar cada árvore para o total de sítios, será considerada como sendo a escolha final do processo (Weir, 1996).

Em geral, os métodos baseados em parcimônia operam selecionando as árvores de menor comprimento total, dado pelo número de passos evolucionários (transformações de um caractere de um estado para outro) requeridos para explicar um conjunto de dados. De uma maneira mais formal, pode-se definir o problema geral da máxima parcimônia da seguinte maneira. A partir do conjunto de todas as possíveis árvores, deve-se encontrar todas as árvores τ tal que

$$L(\tau) = \sum_{k=1}^{B} \sum_{j=1}^{N} w_j * diff(x_{k'j}, x_{k''j})$$

seja mínimo, onde $L(\tau)$ é o comprimento da árvore τ, *B* é o número de ramos, *N* é o número de caracteres, *k*' e *k*" são dois nós incidentes em cada ramo *k*, $x_{k'j}$ e $x_{k''j}$ representam cada elemento da matriz de dados de entrada e diff(*y*,*z*) é a função que especifica o custo de

transformação do estado *y* para o estado *z* ao longo de cada ramo. O coeficiente w_j associa um peso a cada caractere (Swofford *et al.*, 1996).

Máxima Parcimônia Ponderada e não Ponderada

Nei & Kumar (2000) afirmam que é possível dividir as muitas versões de métodos baseados em Máxima Parcimônia em dois grupo: Máxima Parcimônia Ponderada e Máxima Parcimônia não Ponderada. Nos métodos não ponderados, as substituições de nucleotídeos ou de aminoácidos podem ocorrer em todas as direções com a mesma probabilidade.

No entanto, é sabido que as substituições de transição (α) ocorrem com maior freqüência do que as de transversão (β). Segundo Swofford *et al.* (1996), a proporção entre transição e transversão é $R = \frac{\alpha}{2\beta}$. A Figura 3.8 ilustra estes tipos de alterações.

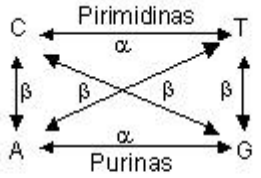


Figura 3.8 Taxas de transição (α) e de transversão (β) de Kimura (Swofford *et al.*, 1996)

Por isto, parece ser razoável dar pesos diferentes para alterações diferentes. Os métodos que incorporam esta característica são os do tipo Máxima Parcimônia Ponderada. É esperado que estes métodos produzam topologias mais confiáveis que os outros (Nei & Kumar, 2000).

Os métodos baseados em Máxima Parcimônia ignoram informações sobre os comprimentos dos ramos das árvores durante o processo de busca da melhor árvore, justamente por não recorrerem a nenhum modelo evolutivo, o qual pode levar a uma estimativa do comprimento dos ramos (Swofford *et al.*, 1996).

3.7 Métodos baseados em modelo

São métodos que requerem modelos explícitos para sua formulação (Swofford *et al.*, 1996).

3.7.1 Máxima Verossimilhança

Os métodos baseados em Máxima Verossimilhança para construção de árvores filogenéticas evitam as limitações dos outros métodos; entretanto eles requerem capacidade computacional muito maior para seu processamento. Ao contrário dos métodos baseados em Matriz de Distâncias, os métodos baseados em Máxima Verossimilhança tentam usar explícita e eficientemente todas as informações disponíveis, ao invés de reduzi-las a um conjunto de distâncias. Eles são baseados em modelos probabilísticos de evolução (Felsenstein, 1981).

Na forma como são implementados atualmente, os métodos baseados em Máxima Verossimilhança supõem uma topologia para a árvore, e selecionam os comprimentos dos ramos de maneira a maximizar a probabilidade dos dados analisados em função da árvore em estudo. Estas probabilidades são comparadas com as de árvores de outras topologias, e aquela que apresenta a maior probabilidade é considerada a mais verossímil (Nei & Kumar, 2000; Weir, 1996).

O cálculo da Máxima Verossimilhança (Weir, 1996) leva em consideração todos os sítios e todas as possibilidades de mutações em todos os nós internos da árvore proposta. Observando a Figura 3.9, vamos obter a verossimilhança da árvore considerada.

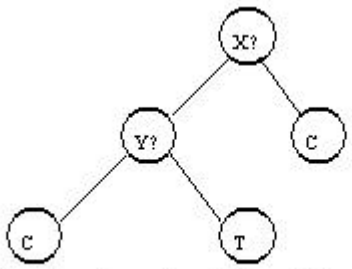


Figura 3.9 Ilustração dos elementos efetivos (folhas) e dos hipotéticos (X e Y)

A Máxima Verossimilhança será o resultado do cálculo de:

$L = L_1 * L_2 * ... * L_n = \prod_{i=1}^{N} L_i$, onde:

N é o número de sítios,

L_i é a verossimilhança para o sítio *i*, calculada levando-se em conta todas as possíveis transições da topologia em estudo.

Os nós externos (folhas da árvore) contêm as bases C, T e C mas não é possível determinar com certeza quais são as bases dos nós internos (ancestrais hipotéticos). O método da Máxima Verossimilhança analisa todas as hipóteses possíveis. Baseado em algum dos modelos de substituição de bases, comentados mais a frente neste capítulo, é atribuída uma probabilidade para cada possível alteração, ou seja, leva-se em consideração a probabilidade da base A não sofrer alteração alguma (permanecer em A), a probabilidade da base A sofrer alteração para a base C e assim sucessivamente até chegar na probabilidade da base T não sofrer alteração alguma (permanecer em T). A Figura 3.10 a seguir ilustra graficamente estas probabilidades.

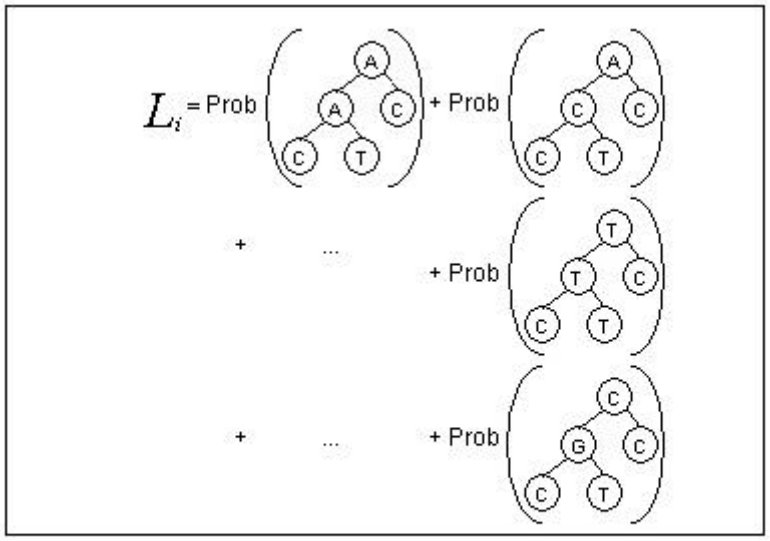


Figura 3.10 A verossimilhança de um sítio em particular é a soma das probabilidades de todas as possíveis reconstruções dos ancestrais, dado algum modelo de substituição de bases (Swofford *et al.*, 1996)

$L_i = \sum_{k=1}^{4}\sum_{l=1}^{4}...\sum_{z=1}^{4}\pi_k P_{kl}(v_i)P_{lj}(v_j)...P_{zn}(v_n)$. L_i é verossimilhança da árvore em estudo, dado um sítio em particular, onde:

$\pi_k = 0{,}25$, segundo o modelo de substituição de Jukes-Cantor (1969) ou é o fator relativo à freqüência das bases nas cadeias estudadas, segundo o modelo de substituição de Felsenstein (1981):

$P_{ij} = (T)$ é a probabilidade da base i mudar para a base j após o tempo T.

$P_{ii} = (1 - p_i) + p_i\pi_i$, é a probabilidade da base *i* não sofrer alteração alguma.

$P_{ij} = p_i\pi_j$; $i \neq j$, é a probabilidade da base *i* sofrer alteração para a base *j*.

$p_i \cong 1 - e^{-v_i}$, onde v_i representa o relógio molecular ou o comprimento do ramo *i*.

Como os valores de L tendem a ser muito pequenos, é comum o uso do logaritmo natural de *L*, conforme indicado a seguir:

$$\ln L = \ln L_1 + \ln L_2 + ... + \ln L_N = \sum_{i=1}^{N} \ln L_i$$

A seguir serão apresentados três modelos de substituição de bases mais comuns para análise da Máxima Verossimilhança.

Comentários sobre os modelos substituição de bases de Jukes-Cantor (1969), Kimura (1980) e Felsenstein (1981)

Se assumirmos que as freqüências de equilíbrio de todas as bases são iguais, ($\pi_A = \pi_C = \pi_G = \pi_T = 0{,}25$) e que todas as substituições ocorrem na mesma taxa ($\alpha = \beta$, conforme ilustrado na Figura 3.8), então as freqüências de equilíbrio e taxas de substituição podem ser combinadas em um único parâmetro. Este modelo é o mais simples de todos e é conhecido como modelo de Jukes-Cantor (1969) e normalmente é referenciado de maneira abreviada por "JC-1969" (Swofford *et al*., 1996).

$$\begin{bmatrix} -3\alpha & \alpha & \alpha & \alpha \\ \alpha & -3\alpha & \alpha & \alpha \\ \alpha & \alpha & -3\alpha & \alpha \\ \alpha & \alpha & \alpha & -3\alpha \end{bmatrix}$$

O modelo de Kimura (1980) leva em consideração o fato de que transições (α) e transversões (β) ocorrem em taxas diferentes (veja Figura 3.8), mas ainda assume freqüências de bases iguais (Swofford *et al*., 1996). Se considerarmos $\alpha = \beta$, este modelo se converte no modelo proposto por Jukes-Cantor (1969).

$$\begin{bmatrix} -\alpha - 2\beta & \beta & \alpha & \beta \\ \beta & -\alpha - 2\beta & \beta & \alpha \\ \alpha & \beta & -\alpha - 2\beta & \beta \\ \beta & \alpha & \beta & -\alpha - 2\beta \end{bmatrix}$$

Felsenstein (1984) usou um método diferente para acomodar freqüências de bases diferentes em um modelo de dois parâmetros que hoje é conhecido como "F84". Seu modelo divide o processo de substituição em dois componentes: uma taxa de substituição geral capaz de produzir todos os tipos de substituições, e uma taxa de substituição interna que produz somente transições (Swofford *et al*., 1996), produzindo:

$$\begin{bmatrix} - & \mu\pi_C & \mu\pi_G(1+K/\pi_R) & \mu\pi_T \\ \mu\pi_A & - & \mu\pi_G & \mu\pi_T(1+K/\pi_Y) \\ \mu\pi_A(1+K/\pi_R) & \mu\pi_C & - & \mu\pi_T \\ \mu\pi_A & \mu\pi_C(1+K/\pi_Y) & \mu\pi_G & - \end{bmatrix}$$

onde K é o parâmetro que determina a taxa de transições:transversões, $\pi_R = \pi_A + \pi_G$, $\pi_Y = \pi_C + \pi_T$ e os elementos da diagonal principal são a soma dos elementos já definidos de cada linha, multiplicados por –1.

De acordo com Swofford *et al*. (1996) e com Weir (1996), este modelo pode ser reduzido ao de Kimura (1980) quando todos os valores de π são iguais, e também pode ser reduzido ao modelo de Felsenstein (1984) quando as diferenças entre transições e transversões são ignoradas ($\alpha = \beta$). Quando estas duas simplificações são utilizadas, resulta o modelo de Jukes-Cantor (1969). A matriz a seguir representa a composição dos três modelos de substituição estudados e a partir dela é possível obter qualquer um dos três modelos fazendo as simplificações explicadas.

$$\begin{bmatrix} -\beta\pi_C - \alpha\pi_G - \beta\pi_T & \beta\pi_C & \alpha\pi_G & \beta\pi_T \\ \beta\pi_A & -\beta\pi_A - \beta\pi_G - \alpha\pi_T & \beta\pi_G & \alpha\pi_T \\ \alpha\pi_A & \beta\pi_C & -\alpha\pi_A - \beta\pi_C - \beta\pi_T & \beta\pi_T \\ \beta\pi_A & \alpha\pi_C & \beta\pi_G & -\beta\pi_A - \alpha\pi_C - \beta\pi_G \end{bmatrix}$$

Conclusões

Ao longo deste capítulo, foi realizada uma breve revisão e discussão sobre os principais conceitos necessários para o entendimento do problema das árvores filogenéticas em geral e dos principais métodos disponíveis para sua reconstrução. A ferramenta computacional desenvolvida ao longo desta pesquisa foi baseada nos métodos da Matriz de Distâncias e da Máxima Verossimilhança. O uso dos outros métodos fica como sugestão para extensão do presente trabalho.

Capítulo 4

Análise Filogenética

4.1 Introdução

Um ponto em comum identificado nos trabalhos estudados a seguir é o reconhecimento da dificuldade de ser encontrada uma boa solução dentro do espaço de busca das árvores candidatas. Todos deixam claro que busca exaustiva é inviável quando a quantidade de elementos estudados passa de uma ou duas dezenas. Para apenas 14 elementos, o número de árvores candidatas chega a sete trilhões e para 50, este número salta para a casa de 2,75 * 10^{76} possibilidades. Para maiores detalhes, veja apêndice B.

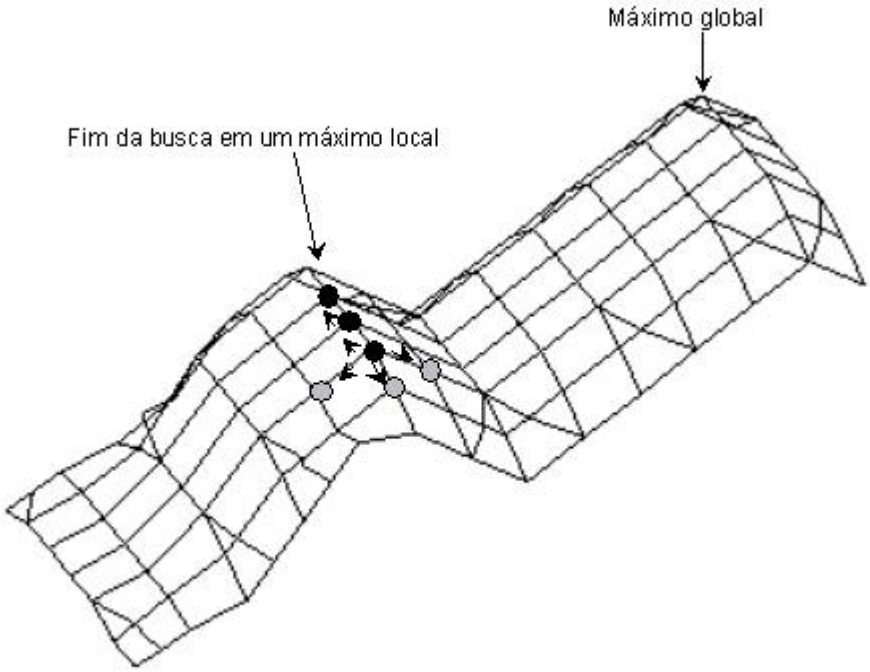


Figura 4.1 A superfície acima apresenta uma visão pictórica de um espaço de solução bi-dimensional para o problema de encontrar uma boa árvore filogenética. Usar apenas busca local, como *hillclimbing*, pode levar à falha ilustrada nesta figura (baseado em Felsenstein, 1997).

Uma técnica comum é tomar uma estimativa inicial de uma árvore, e fazer pequenos rearranjos em seus ramos, de modo a alcançar as árvores "vizinhas". Se qualquer uma destas vizinhas for melhor que a original, ela passa a ocupar o posto de referência no lugar da árvore original, e o processo de rearranjos prossegue até que não seja mais possível encontrar outra árvore melhor do que a que ocupa o posto de referência. A árvore encontrada ocupa uma posição de ótimo local no espaço de árvores. Entretanto, não existem garantias que esta seja também uma posição de ótimo global. A figura 4.1 representa uma ilustração do problema, para o caso de busca em espaço de coordenadas bi-dimensional. Neste diagrama, a busca foi iniciada em um ponto da superfície, e depois todas as árvores vizinhas foram examinadas (neste exemplo são quatro). Uma delas é melhor que a original, então ela passa a ocupar o posto de referência. O processo de analisar os vizinhos continua até que se alcance o ponto mais alto desta "montanha" (*hill*). Como pode ser observado neste diagrama, esta estratégia é incapaz de escapar dos máximos locais (Felsenstein, 1997).

Uma grande vantagem de se usar computação evolutiva é a possibilidade de se trabalhar com muitos pontos de partida simultaneamente, além da disponibilidade de mecanismos de recombinação das soluções existentes. Com isso aumentam-se as chances de escapar de máximos locais ruins e atingir o máximo global ou pelo menos um bom máximo local.

A seguir, serão apresentados alguns trabalhos mais recentes que tentam contornar o problema da dimensão do espaço de soluções. As estratégias mais promissoras se concentram em torno do uso combinado de Computação Evolutiva com algum método de busca local.

4.2 Estudo de alguns trabalhos sobre Filogenia

4.2.1 Antes do uso de Algoritmos Genéticos

Matsuda *et al*. (1993)

Neste trabalho de 1993, Matsuda e outros se mostraram muito preocupados com o elevado custo computacional imposto pelo método da Máxima Verossimilhança, apesar de acreditarem que este método seja superior aos outros por fazer uso explícito de todos os dados disponíveis para análise e por se basear em um modelo evolutivo. Os autores afirmaram que o custo de processamento para mais de 20 elementos se torna inviável.

Eles não aceitavam trabalhar com apenas 20 elementos quando já era possível trabalhar com várias centenas que estavam disponíveis em bancos de dados de genes, como por exemplo, GenBank (Burks *et al*., 1992) e EMBL (Higgins *et al*., 1992). Eles buscaram uma

alternativa que superasse as limitações dos algoritmos usados naquela época. O algoritmo estudado foi baseado na idéia proposta por Felsenstein (1990) e Olsen *et al*. (1993), que consiste basicamente em inserir o elemento *i* na árvore já existente contendo i - *1* elementos até que todos os *n* elementos sejam inseridos na árvore.

A maneira proposta para contornar o limite de 20 elementos foi utilizar processamento paralelo, distribuído pelos processadores de um computador, conforme exemplificado na figura a seguir.

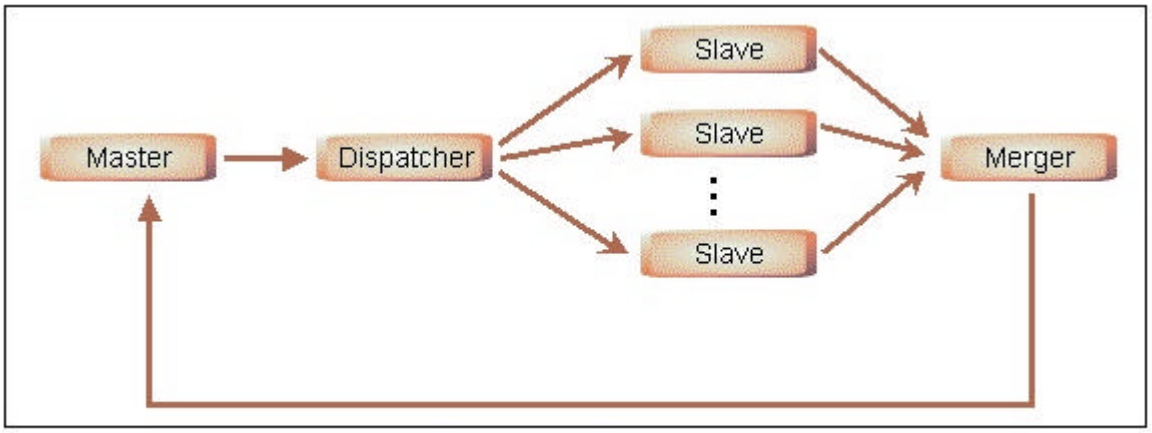


Figura 4.2 Processo para busca de árvores filogenéticas (Matsuda *et al.*, 1993).

O "master" tem função de criar as árvores candidatas, o "dispatcher" se encarrega de distribuir as árvores criadas para os "slaves", que por sua vez têm a função de calcular os comprimentos dos ramos e verossimilhança da árvore. O "merger" se encarrega de recolher as árvores, os comprimentos dos ramos e os índices calculados para serem enviados de volta ao "master". É muito interessante a maneira como esta idéia se assemelha ao princípio geral dos algoritmos genéticos, onde os candidatos gerados e analisados segundo a função de *fitness* se assemelham ao trabalho realizado pelos "slaves" e "merger". Aqui, poderia estar a semente que viria a gerar os trabalhos de Matsuda de 1995 e 1996, onde ele já propõe o uso de algoritmos genéticos para o estudo das árvores filogenéticas.

4.2.2 Primeira referência sobre Algoritmos Genéticos para o problema das árvores filogenéticas

Matsuda (1996)

Neste trabalho de 1996, Matsuda descreve seu método de construção e análise de árvores filogenéticas baseado em algoritmos genéticos. Uma grande diferença de seu método

em relação aos algoritmos genéticos tradicionais da época foi o uso de codificação do DNA em forma de grafo, contra os binários normalmente utilizados. Outra grande contribuição de seu método foi o esquema criado para realizar o *crossover* de duas árvores existentes para formar uma nova, baseada em um critério dividido em três passos. O fato de ele ter usado árvores sem raiz tornou o *crossover* menos complicado do que se ele estivesse usando árvores com raiz, pois seriam necessários muitos ajustes para evitar a geração de árvores desconexas.

Neste trabalho, Matsuda não comenta sua idéia apresentada em 1993 que consistia em dividir o trabalho de cálculo dos comprimentos dos ramos e do valor da Máxima Verossimilhança pelos vários processadores de um computador paralelo. Parece ser interessante empregar vários processadores associado ao uso de algoritmo genético. O trabalho de analisar os vários candidatos gerados poderia ser distribuído entre os processadores paralelos e assim poderia ser acelerado o processo de busca.

A comparação realizada por Matsuda apresenta este método como sendo igual ou superior aos métodos existentes em sua época.

4.2.3 Lewis (1998)

Neste trabalho, os princípios dos algoritmos genéticos são explicados mais detalhadamente do que nos trabalhos anteriores e de outros autores.

É comentada a impossibilidade de serem analisadas todas as árvores candidatas quando o número de indivíduos envolvidos se encontra em torno de algumas centenas. Nestes casos, é necessário usar alguma heurística para acelerar o processo e não se pode esperar encontrar o melhor elemento mas algum bom elemento como resposta para o conjunto de elementos estudados.

Outra observação interessante encontrada neste trabalho é que, embora os algoritmos genéticos se baseiem muito em conceitos de evolução, seu uso para resolver problemas de evolução e outras áreas da biologia está apenas começando.

É justificado o uso de algoritmos genéticos para o problema das árvores filogenéticas por causa de sua habilidade de encontrar soluções quase ótimas rapidamente, mesmo quando muitos elementos estão envolvidos, ou modelos evolucionários complexos são utilizados, como é o caso da Máxima Verossimilhança. Outro ponto forte é o paralelismo implícito que permite analisar DNAs de centenas de indivíduos simultaneamente. Os computadores com vários processadores paralelos poderão acelerar o processo de busca. A possibilidade de distribuir os

processos de busca em relação aos sítios pelos processadores pode vir a permitir que centenas ou milhares de elementos possam ser analisados simultaneamente no futuro.

Uma diferença interessante neste método em relação aos anteriores é que, no início do processo de busca, ele usa um valor fixo para os comprimentos dos ramos (por exemplo 0,05) e calcula a verossimilhança dos candidatos sem realizar otimização alguma neste ponto. Esta otimização é o que mais consome recursos computacionais e este é o grande avanço proporcionado por este método. A ferramenta computacional desenvolvida neste trabalho de mestrado (veja Capítulo 5) também faz algo semelhante. Nenhuma otimização é realizada quanto aos comprimentos dos ramos até que a melhor árvore candidata seja encontrada. Neste momento, é realizada a otimização dos comprimentos dos ramos da melhor árvore candidata e em seguida é calculada sua verossimilhança.

4.2.4 Reijmers *et al.* (1999)

Neste trabalho, foi usado algoritmo genético para a otimização de árvores filogenéticas, com o critério de avaliação baseado no método da Matriz de Distâncias.

A estratégia utilizada consiste em determinar a topologia inicial usando o método Neighbor-Joining (Saitou & Nei, 1987), determinar os comprimentos dos ramos de acordo com o método de Fitch & Margoliash (1967) usando a matriz de distâncias original. Após encontrar a topologia inicial e os comprimentos dos ramos, uma nova matriz de distâncias pode então ser deduzida. Pela comparação desta nova matriz de distâncias com a original, uma medida da qualidade pode ser determinada para esta árvore.

Para uma dada matriz de distâncias, a topologia da árvore é corrigida aplicando uma matriz de correções com pequenas alterações em relação aos valores existentes. Os elementos desta matriz de correções são otimizados com o uso de um algoritmo genético. Isto automaticamente fará com que o algoritmo genético execute busca de caráter predominantemente local pela matriz de correções ótima, que resultará em uma árvore com boa topologia.

4.2.5 Skourikhine (2000)

Neste trabalho, o autor informa que foram experimentadas modificações no algoritmo genético padrão, com amplo uso de idéias extraídas dos algoritmos evolucionários para a busca das árvores filogenéticas usando máxima verossimilhança. Em particular, foi usado o conceito

de auto-adaptação de parâmetros do algoritmo genético, diferentes esquemas de seleção, mutação e recombinação, que proporcionaram ganho em eficiência e desempenho.

É esclarecido que a verossimilhança de uma árvore não é a probabilidade de que esta árvore seja correta. Ao contrário, ela é apenas a probabilidade dos dados observados caso a árvore e os comprimentos dos ramos estejam corretos. Os métodos de inferência evolucionária baseados em Máxima Verossimilhança avaliam uma hipótese sobre a história evolucionária que seja a mais consistente com os dados observados nos termos do modelo evolucionário proposto.

Também é informado que, para mais de dez elementos, quando a busca exaustiva usando o método da Máxima Verossimilhança se torna proibitiva, é necessário usar alguma forma de heurística que tentará encontrar uma boa resposta e não a ótima. Um dos métodos mais promissores neste caso é o algoritmo genético.

A idéia básica do algoritmo genético auto-ajustado é partir do controle global do mecanismo usado no algoritmo genético padrão e distribuí-lo pelos indivíduos da população em estudo. Desta forma, também os controles são evoluídos ao longo das gerações e de forma independente para cada indivíduo.

O uso de muitos parâmetros auto-ajustáveis pode ser mais eficiente, pois com seu uso é possível atingir um maior grau de exploração do espaço de solução do problema, em comparação com o uso de apenas um parâmetro auto-ajustável. Entretanto, pode-se também verificar a redução da velocidade de convergência, caso os parâmetros sejam determinados de forma inadequada. Maior robustez pode ser alcançada pela aplicação de tipos especiais de seleção.

4.2.6 Goloboff & Farris (2001)

Neste trabalho, os autores não usaram Máxima Verossimilhança nem algoritmos genéticos, mas foram capazes de obter bons resultados com a estratégia adotada. Esta estratégia é bem similar ao princípio básico dos algoritmos genéticos, por utilizar uma série de buscas simples e depois analisar os resultados destas buscas para criar uma solução mais apropriada em relação ao problema em estudo.

Conforme informado pelo autor, Rice *et al*. (1997) não foram capazes de encontrar a árvore mais adequada para 500 elementos mesmo após 11,5 meses de busca usando PAUP (Swofford, 1993). Em contraste, Nixon (1999) usando NONA (Goloboff, 1994) foi capaz de encontrar as árvores de mínimo comprimento em uma média de quatro horas e Goloboff *et al*.

(1999), usando TNT e estratégias combinadas de busca foram capazes de completar a busca em uma média de dez minutos.

Seu método consiste em comparar os resultados de buscas diferentes e independentes. As árvores encontradas nestas buscas são condensadas, resultando em uma árvore de estrito consenso. O consenso é obtido pela análise das melhores árvores encontradas pelas buscas anteriores. Este método assume que os resultados recuperados pela maioria das buscas são mais prováveis de serem suportados pelos dados.

Apesar de não usarem algoritmos genéticos, os autores geram várias árvores candidatas que são avaliadas por uma regra (regra da maioria) para produzir o que eles chamaram de árvore de estrito consenso.

4.2.7 Janis & Wheeler (2001)

Neste trabalho, os autores também não usaram algoritmos genéticos nem o método da Máxima Verossimilhança, porém eles conseguiram bons resultados devido ao uso de processamento paralelo.

Também é comentado o tremendo avanço que os algoritmos de busca de árvores filogenéticas experimentaram nos últimos dois anos. Estes avanços incluem o uso de algoritmos genéticos, fusão de árvores (Goloboff, 1999), buscas setoriais (Goloboff, 1999) e "simulated annealing" (Nixon, 1999; Goloboff, 1999).

A maneira como o método da Máxima Verossimilhança foi implementado por Janis e Wheeler é um pouco parecida com a que foi usada por Matsuda *et al*. (1993). Um processo principal é utilizado para distribuir as tarefas pelos processos escravos que se encarregam de completar suas tarefas e devolver o resultado para o processo principal. O processo principal recebe os retornos, analisa os resultados e decide sobre os próximos passos, passando em seguida as novas tarefas de volta para os processos escravos. Este ciclo se repete até que uma boa resposta seja encontrada para o problema.

Ao contrário de Matsuda *et al*. (1993), que usou Máxima Verossimilhança, neste trabalho foi usada uma recente adaptação do método da Parcimônia chamada "Ratchet" (Nixon, 1999), que se mostrou muito eficiente para explorar o espaço de soluções das árvores filogenéticas, usando o artifício de analisar múltiplas ilhas de árvores, enquanto que o método tradicional utiliza apenas uma ilha para sua busca.

4.2.8 Moilanen (2001)

Apesar de neste trabalho também não ser abordado o uso do método da Máxima Verossimilhança, importantes resultados são alcançados em termos de tempo de busca e qualidade da resposta encontrada, devido à combinação de algoritmos de busca global (algoritmo genético) com algoritmos de busca local ("*branch swapping*"). Moilanen utilizou em sua pesquisa uma variação do método da Parcimônia chamada "Ratchet" (Nixon, 1999) para a busca global e "*branch swapping*" para as buscas locais.

Conforme demonstrado, os algoritmos de busca local são muito rápidos e eficientes para encontrar soluções, porém não conseguem escapar dos ótimos locais. Os algoritmos de busca global são capazes de escapar dos ótimos locais, porém necessitam de maior tempo de processamento do que os anteriores. Por isto, parece que a combinação destes dois tipos de algoritmos constitui uma forma de busca muito poderosa pela união das boas qualidades dos dois tipos primitivos.

Neste trabalho, também é comentado a respeito dos prováveis ganhos com uso de processamento paralelo, distribuindo as cargas de processamento por vários processadores. Parece que esta opinião vem se tornando cada vez mais popular nos últimos anos.

Segundo este autor, os métodos híbridos que combinam algoritmos de busca global com algum algoritmo de busca local vêm sendo usados para resolver vários problemas, e em todos estes estudos foi constatado que esta combinação era mais eficiente do que os métodos primitivos separados.

Resultados obtidos como Projeto Árvores Filogenéticas

Ainda não foi possível realizar comparações diretas entre os resultados obtidos com o "Projeto Árvores Filogenéticas" com os resultados obtidos com outras ferramentas. Alguns autores estudados apresentam seus resultados, mas não fornecem as seqüências de bases utilizadas em suas pesquisas. Outros apresentam algumas seqüências de bases, mas trabalham com árvores sem raiz. Até o momento, não foi possível reunir todas as características necessárias para comparações diretas.

A tabela a seguir mostra os resultados obtidos para três conjuntos de seqüências hipotéticas que foram submetidas a três ferramentas para análise filogenética: PAML (Yang, 1999), Phylip (Felsenstein, 1990) e o Projeto Árvores Filogenéticas.

	4 seqüências	5 seqüências
Phylip	-130,74914	-163,87052
Projeto Árvores Filogenéticas	-136,14495	-171,11214
PAML	-136,19721	-174,34676

Como o método da Máxima Verossimilhança busca maximizar a probabilidade de uma árvore dada uma seqüência de bases, os maiores valores serão os melhores. Em todas as colunas da tabela acima os resultados obtidos com Phylip de Felsenstein foram os melhores por serem sempre os maiores. Uma leitura rápida pode confundir os leitores que não atentarem para o fato destes resultados serem todos negativos. Levando em consideração que as topologias das árvores utilizadas pelas ferramentas Phylip e PAML não possuem raiz, seria normal esperar que seus valores de Máxima Verossimilhança fossem melhores do que o obtido com o Projeto Árvores Filogenéticas onde são utilizadas árvores com raiz e por isto, existe uma quantidade maior de ramos a serem considerados nos cálculos.

Os resultados são apenas próximos e nem poderiam ser iguais, pois as três ferramentas apresentam características diferentes. O Projeto Árvores Filogenéticas analisa todas as bases das seqüências e trabalha com árvores com raiz, enquanto que PAML apenas analisa as bases nos sítios onde exista alguma diferença entre as seqüências e trabalha com árvores sem raiz; Phylip analisa todas as bases das seqüências mas trabalha com árvores sem raiz.

Conclusões

Conforme observado nos trabalhos estudados anteriormente, a idéia de usar algoritmos genéticos para o problema das Árvores Filogenéticas está se solidificando como útil, prática e robusta. Apesar de seu início relativamente recente (Matsuda, 1995), esta estratégia vem sendo explorada cada vez mais intensamente com o passar dos anos. A quantidade de trabalhos publicados em 2001 aumentou significativamente em relação aos anos anteriores.

A tendência mais recente é usar os algoritmos genéticos associados a algum tipo de busca local. Desta forma é possível unir as boas características das duas estratégias: a capacidade de evitar os máximos locais dos algoritmos genéticos unida com a rapidez e agilidade dos métodos de busca local.

As obras utilizadas ao longo desta pesquisa de mestrado não forneceram informações detalhadas sobre todos os passos necessários para a solução do problema de reconstruir uma

boa árvore filogenética. Este é um grande diferencial deste trabalho em relação aos demais pesquisados. Aqui se procurou fornecer visão holística do problema de reconstrução das árvores filogenéticas, desde a análise das seqüências de bases até o ajuste dos comprimentos dos ramos, que foi implementado baseado em um método numérico de primeira ordem conhecido como “Método do Gradiente”.

Outro ponto que merece ser destacado é o uso de técnicas de Computação Evolutiva, que se mostrou uma poderosa ferramenta para ajudar no problema de encontrar uma boa árvore filogenética, a partir de uma seqüência de bases analisadas, devido ao seu elevado poder de exploração e explotação de vastos espaços de solução.

Também foi apresentado o uso de codificação do DNA (usado pelo algoritmo genético) baseado em grafo, por ser capaz de preservar as informações relativas à hierarquia (ancestrais e descendentes) em relação às estruturas tradicionalmente utilizadas, que normalmente são baseadas em listas e preservam apenas informações lineares.

Novos operadores especiais de mutação para o algoritmo genético foram propostos, baseados em trocas de colunas da matriz de adjacências do grafo utilizado para representar a árvore filogenética candidata. Uma nova ferramenta computacional (Projeto Árvores Filogenéticas) foi desenvolvida e disponibilizada para outros pesquisadores contendo as seguintes características:

- Possui interface gráfica amigável, projetada especialmente para facilitar seu uso por usuários que não dominem completamente as técnicas de computação;
- Permite ajuste dos principais parâmetros relativos a filogenia, tais como seleção do método de busca a ser utilizado (Matriz de Distâncias ou Máxima Verossimilhança) e modelo de substituição de bases (Jukes-Cantor, 1969; Felsenstein, 1981 ou Kimura, 1980);
- Permite ajuste dos principais parâmetros relativos ao algoritmo genético utilizado, tais como tamanho da população e porcentagem de mutações.

Capítulo 5

Resultados Experimentais da Pesquisa

5.1 Introdução

Ao longo deste trabalho de mestrado procurou-se ratificar que a Computação Evolutiva é uma ferramenta muito poderosa para auxiliar na busca de boas árvores filogenéticas. Uma grande dificuldade que impede o tratamento deste problema com técnicas convencionais é a dimensão do espaço de soluções que cresce em escala fatorial. Como já foi exposto anteriormente, é uma pretensão muito grande tentar encontrar a melhor árvore dentro de um espaço de solução gigantesco, que pode facilmente atingir a casa de $2{,}75 * 10^{76}$ possibilidades, quando são estudados os DNAs de somente 50 indivíduos que comporão as folhas desta árvore (Weir, 1996). Neste cenário, os algoritmos genéticos vêm se mostrando como sendo uma alternativa muito boa para auxílio aos pesquisadores por causa de seu elevado poder de exploração e explotação (Bäck *et al.*, 1997).

Técnicas de otimização com características de busca global, como aquelas que trabalham com uma população de candidatos à solução a cada iteração, se mostraram eficazes em muitos problemas em que uma boa solução deve ser encontrada dentre uma enorme quantidade de candidatos, apesar de não oferecerem garantias de encontrar a solução ótima (Massart *et al.*, 1997). Uma das técnicas de otimização global mais freqüentemente utilizadas é a dos algoritmos genéticos (Holland, 1973; Goldberg, 1989; Holland, 1992).

Algoritmos genéticos já foram utilizados para otimização de árvores filogenéticas por Matsuda (1995), onde ele empregou algoritmos genéticos na busca de árvores filogenéticas construídas a partir de seqüências de aminoácidos. Em outro trabalho sobre este assunto, Matsuda (1996) também usou algoritmos genéticos para encontrar as árvores filogenéticas, baseado no método da Máxima Verossimilhança.

A ferramenta computacional que foi criada para os testes ao longo desta pesquisa adota a estratégia de evoluir muitas árvores candidatas simultaneamente. Este é o maior trunfo dos algoritmos genéticos para resolver problemas deste tipo, pois existe uma quantidade muito

grande de possíveis caminhos a serem pesquisados. É impossível determinar em princípio qual dos caminhos levará até a árvore de melhor qualidade. Para reduzir as chances de uma busca inútil, a melhor alternativa é analisar vários deles em paralelo.

A parte fundamental da ferramenta computacional é constituída pelo módulo dos algoritmos genéticos. O algoritmo pode ser ajustado para trabalhar somente com o modelo da Matriz de Distâncias, somente com o modelo da Máxima Verossimilhança ou com ambos, situação em que ele apresenta seu melhor desempenho. Com esta última opção, várias gerações de árvores candidatas são analisadas usando o método da Matriz de Distâncias e somente quando a árvore mais promissora for encontrada sistema ajusta os comprimentos de seus ramos baseado nas fórmulas de Weir (1996), descritas mais adiante neste capítulo. Este programa é capaz de trabalhar com os modelos de substituição de bases de três grandes autores: (a) Jukes-Cantor (1969), onde a probabilidade de ocorrer mutação (π) é a mesma para as quatro bases hidrogenadas (A, G, C e T), ou seja, 25%; (b) Felsenstein (1981), onde a probabilidade de ocorrer mutação (π) é equivalente à quantidade atualmente encontrada nas amostras e; (c) Kimura (1980), onde a probabilidade de ocorrer mutação (π) é a mesma para as quatro bases, como em Jukes-Cantor (1969), porém Kimura introduz dois novos fatores: α para as alterações dentro dos mesmos tipos de bases (Pirimidinas para Pirimidinas ou Purinas para Purinas) e β para as alterações de um tipo para outro (Pirimidinas para Purinas ou Purinas para Pirimidinas). A Figura 5.1 ilustra graficamente estas transições de bases. Esta figura já foi apresentada no Capítulo 3 com o título “Figura 3.8” e está sendo repetida aqui apenas para facilitar a leitura do trabalho.

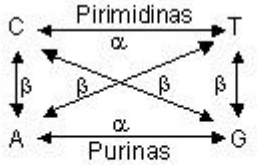


Figura 5.1 Taxas de transição (α) e de transversão (β) de Kimura (Swofford *et al.*, 1996)

Como recurso adicional, o programa é capaz de comparar e exibir os resultados obtidos com os três modelos, baseado no princípio LRT (*Likelihood Ratio Test*) descrito por Weir (1996). Maiores informações sobre esta ferramenta podem ser obtidas no Apêndice A.

5.2 Estudo de árvores filogenéticas usando Computação Evolutiva

Observações sobre o Modelo Evolucionário

Segundo Nei & Kumar (2000), a estrutura básica de todos os seres vivos é escrita com Ácido Desoxirribonucléico (DNA). Por causa disto, pode-se estudar o relacionamento evolucionário entre organismos pela comparação de seu DNA. Esta maneira apresenta várias vantagens em relação à maneira clássica, na qual características morfológicas e fisiológicas são usadas:

- Primeiro, o DNA é composto por quatro tipos de nucleotídeos, adenina (A), timina (T), citosina (C) e guanina (G), e pode ser usado para comparar qualquer grupo de organismos, inclusive bactérias, plantas e animais;
- Segundo, como as alterações evolucionárias no DNA seguem um padrão regular, é possível usar um modelo matemático para comparar DNAs de organismos pouco relacionados;
- Terceiro, o genoma de todos os organismos consiste de longas seqüências de nucleotídeos e contém quantidade muito maior de informação filogenética do que poderia ocorrer no estudo de características morfológicas.

Por estas razões, é esperado que a filogenia molecular venha a esclarecer muitos padrões de ramos da árvore da vida que têm sido difíceis de resolver com métodos baseados em características morfológicas e fisiológicas. Não se pode ignorar, no entanto, a existência de uma dificuldade intrínseca: dada uma seqüência de nucleotídeos, não se constitui uma tarefa simples a atribuição de grau de funcionalidade a cada elemento da seqüência, que conduziria a uma análise filogenética ponderada. O que se faz geralmente, e também será adotado aqui, é atribuir o mesmo grau de funcionalidade a todos os elementos da seqüência de nucleotídeos.

Devido às dificuldade e limitações apresentadas, é esperado que as três alternativas existentes hoje (baseadas em Filogenia, Morfologia e Fisiologia), vão coexistir complementando umas às outras.

O que se conclui, portanto, é que todos os métodos já propostos para realizar análise filogenética vão subsistir e devem ser considerados conjuntamente, sempre que possível. As vantagens e desvantagens de cada método devem ser devidamente investigadas, principalmente na ausência de consenso entre as respostas fornecidas para um dado conjunto de dados de entrada.

O modelo mais aceito hoje para estudo das árvores filogenéticas é o baseado na Teoria da Evolução de Darwin. Este modelo propõe que todos os seres atuais descendem de

organismos em comum. Se voltarmos o suficiente no passado, chegaremos a um único organismo que conseguiu copiar a si próprio, dando início a processos primitivos de reprodução.

Conforme explicado em Nei & Kumar (2000), a reprodução nem sempre é fiel e a cópia pode conter alterações em relação ao original (*crossover* e/ou mutação; transversão e/ou transcrição).

A taxa de erro na cópia, no entanto, é muito baixa. É necessário grande intervalo de tempo e condições apropriadas para que uma espécie sofra alterações suficientes para produzir outras espécies.

Baseado nesta teoria é possível afirmar que as espécies atuais não descendem umas das outras. Elas provavelmente surgiram em algum momento no passado de outras espécies em comum e estão gerando outras novas com o passar do tempo, caso fatores vinculados à seleção natural estejam atuando de forma significativa.

Um exemplo: Considere o organismo A1, com DNA AACCGGTT. Ele gerou dois descendentes, o B1, com DNA ATCCGGTT, e o B2, com DNA AACGGGTT. O elemento B1 gerou os descendentes C1, com DNA ATCCGTTT, e o C2, com DNA ATTCGGTT. A árvore filogenética que melhor representa estes elementos é apresentada na Figura 5.2, a seguir:

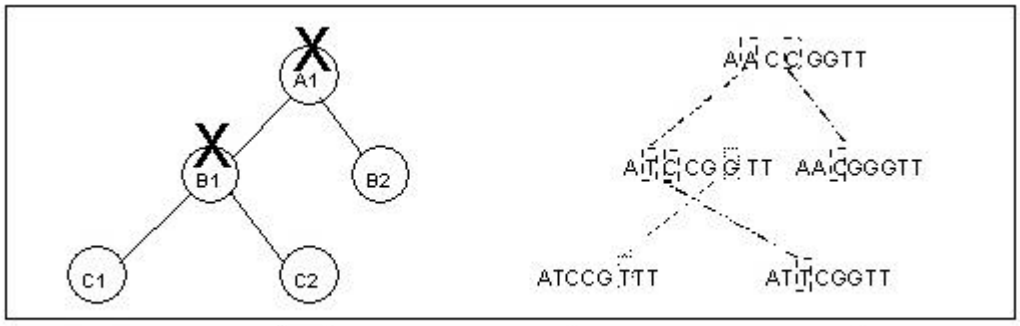


Figura 5.2 Representação gráfica dos DNAs em análise e dos elementos hipotéticos (A1 e B1)

A estratégia de busca para encontrar a melhor árvore filogenética para estes elementos deve iniciar com a análise dos DNAs dos elementos atuais, ou seja, as folhas da árvore. Considerando-se que a taxa de alterações genéticas é baixa, parece coerente propor que os elementos com DNA mais parecidos devem estar mais próximos nesta árvore, e aqueles com mais diferenças devem se encontrar mais distantes.

Os elementos A1 e B1 são ancestrais que não mais existem (ou mesmo que existam não há conhecimento *a priori* para supor que sejam ancestrais de outras espécies) e seu DNA, neste caso, não pode ser determinado com absoluta certeza.

Como a probabilidade de não ocorrer alterações genéticas é maior que a de ocorrer, é fácil perceber que os elementos C1 e C2 da Figura 5.3 a seguir estão mais próximos um do outro do que o elemento C1 de B2 ou C2 de B2, pois a probabilidade da primeira combinação é maior que as outras duas, uma vez que os primeiros contêm mais bases em comum que os demais, para a seqüência de bases em estudo.

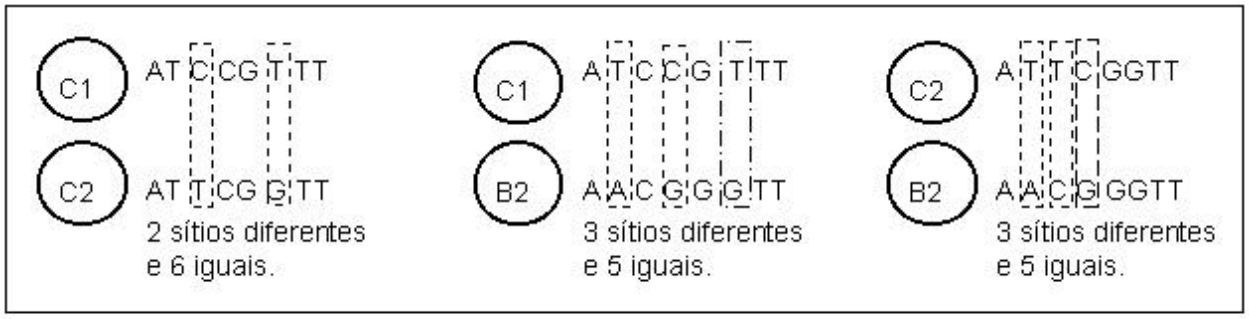


Figura 5.3 Diferenças ressaltadas entre os pares de DNAs analisados.

Encontrar a árvore que melhor represente a situação acima é nitidamente um problema do tipo NP-Completo (Day, 1987) e a quantidade de árvores candidatas para solução do problema pode ser obtida com a seguinte fórmula (Nei & Kumar, 2000) (vide Apêndice B para maiores detalhes)

$$\frac{(2n-3)!}{2^{n-2}(n-2)!}$$

Como pode ser observado, para n = 3, 5 e 9, temos três, 105 e 2.027.025 árvores sem raiz, respectivamente. Ainda não existe nenhuma solução eficiente para esta classe de problemas. Isto significa que, mesmo usando algum critério de otimização relativamente rápido (por exemplo, máxima parcimônia), será necessário tempo excessivamente longo para avaliar todas as árvores candidatas quando for necessário analisar os DNAs de algumas centenas de elementos (Lewis, 1998). A Figura 5.4, baseada em Nei & Kumar (2000), ilustra a quantidade de topologias possíveis para quatro folhas.

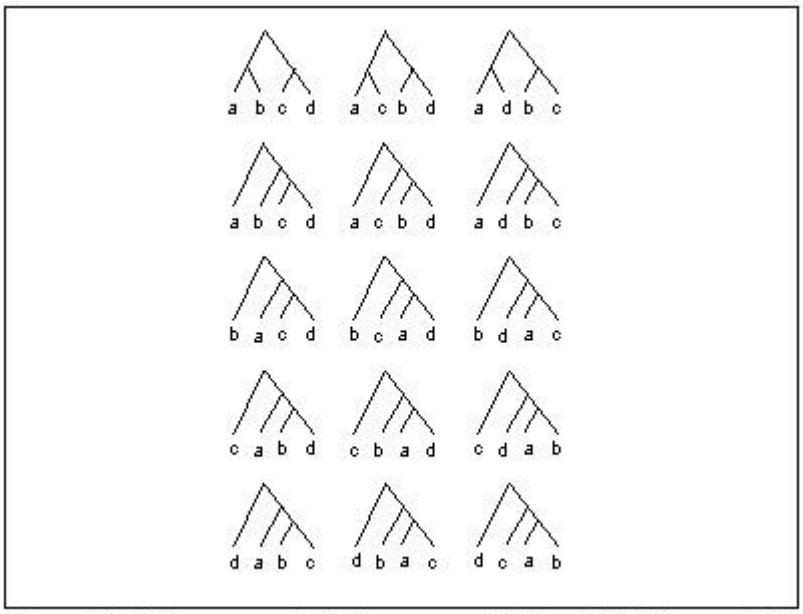


Figura 5.4 Todas as possíveis árvores para 4 folhas (Nei & Kumar, 2000)

O gráfico a seguir apresenta, com escala logarítmica no eixo y, os valores para os primeiros 50 elementos desta série.

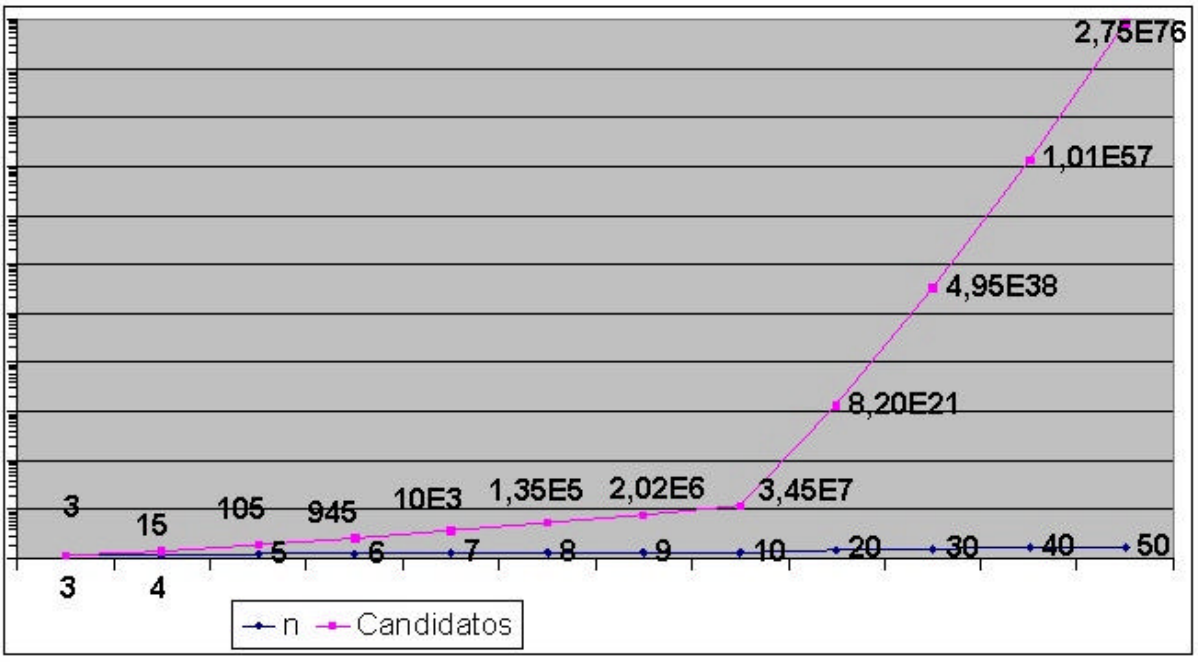


Figura 5.5 Quantidade de árvores candidatas em função das folhas (n). Para maiores informações, veja Apêndice B.

Observações sobre o método da Matriz de Distâncias (não baseado em modelo)

O método da Matriz de Distâncias (Nei & Kumar ,2000) calcula as diferenças entre os DNAs dos elementos estudados e atribui um valor para cada diferença encontrada entre os pares de elementos. Quanto mais bases iguais nos mesmos sítios genéticos encontrados, menor será a distância entre os elementos. A árvore filogenética será montada baseada nas distâncias encontradas entre cada par de elementos. Como este método não pretende calcular o tamanho dos ramos e trabalha apenas com uma matriz de números, ele requer menos processamento que o método da Máxima Verossimilhança, porém ele não apresenta todas as informações que aquele método é capaz de disponibilizar, além do resultado depender do tipo de norma empregada no cálculo da distância.

Conforme descrito anteriormente, neste método são calculadas as distâncias dos DNAs de cada sítio e somadas as distâncias encontradas para cada par de elementos. Estas distâncias são tratadas de forma matricial e usadas para encontrar a melhor árvore para representar os elementos estudados.

Os elementos com menos diferenças são considerados mais próximos geneticamente do que aqueles que apresentam mais diferenças. Deste modo, supõe-se que os elementos com mais diferenças estão mais distantes na linha da evolução das espécies. Na presença de distâncias idênticas, algum critério auxiliar de decisão deve ser adotado. Para efeito desta pesquisa, foi adotado o critério de usar o primeiro elemento dentre todos os que apresentem o mesmo valor.

Os gráficos a seguir exibem os resultados de simulações realizadas com a ferramenta computacional desenvolvida para esta pesquisa. Em todos os casos, foram utilizadas seqüências de DNAs hipotéticas de comprimento curto (por exemplo, dez bases) e o método da Matriz de Distâncias. A ferramenta computacional, baseada em algoritmos genéticos, foi capaz de encontrar uma boa árvore filogenética dentre todas as possibilidades contidas no espaço de solução, em todos os testes efetuados.

Foram necessárias duas figuras apenas por uma questão de escala. Os valores iniciais desapareceriam se todos os valores fossem exibidos no mesmo gráfico.

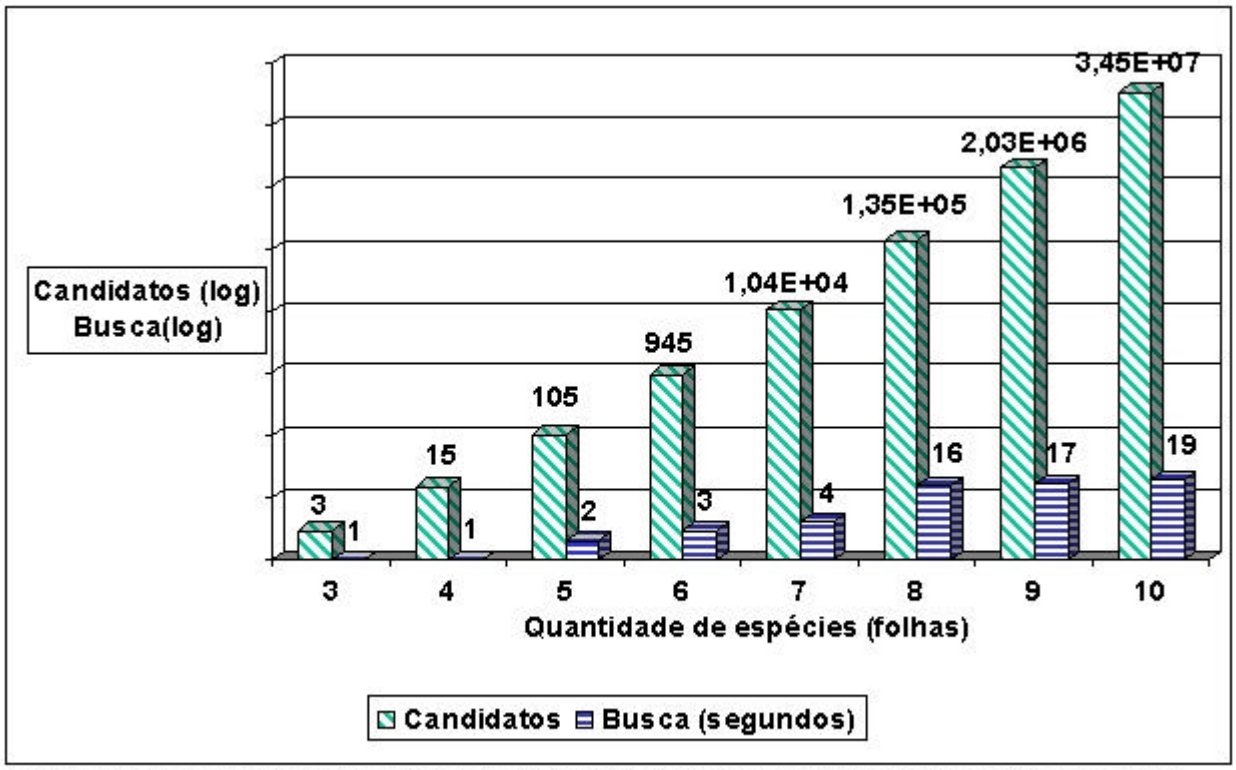


Figura 5.6a Relação entre quantidade de candidatos existentes no espaço de busca e tempo necessário para encontrar uma boa árvore filogenética dentro do intervalo de 1 a 10 seqüências de bases.

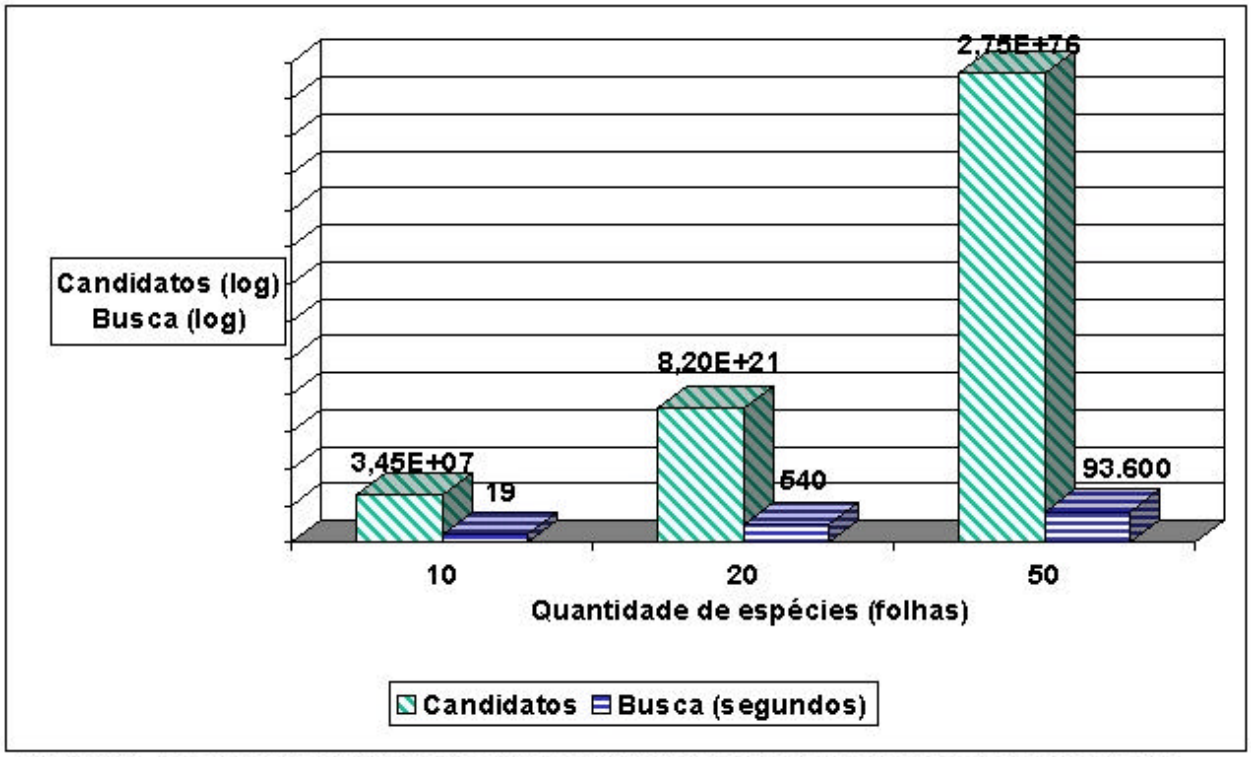


Figura 5.6b Relação entre quantidade de candidatos existentes no espaço de busca e tempo necessário para encontrar uma boa árvore filogenética dentro do intervalo de 10 a 50 seqüências de bases.

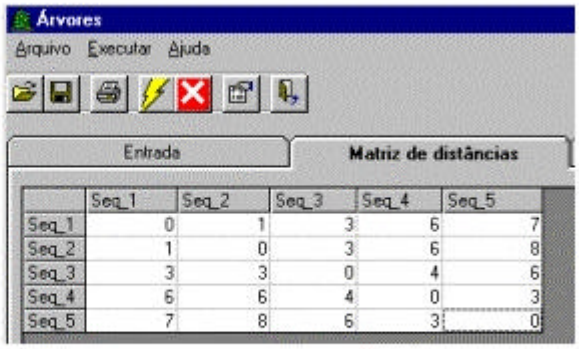
Foi utilizado um microcomputador com processador Intel Pentium II de 300 Mhz com 128 megabytes de memória RAM.

Como pode ser observado no segundo gráfico, o tempo necessário para encontrar uma boa árvore filogenética para 50 elementos foi de aproximadamente 26 horas. Este resultado é aceitável considerando-se que o tempo necessário para efetuar busca exaustiva trabalhando com 25 elementos ou mais é proibitivo (Reijmers e*t al.*, 1999).

Nas figuras a seguir serão exibidas algumas telas da ferramenta computacional, capturadas durante os processamentos de algumas seqüências de bases.

Nas figuras 5.7 a 5.9, serão exibidos os resultados obtidos referentes a cinco seqüências de DNAs.

Seq_1
GTAAATGACA
Seq_2
ATAAATGACA
Seq_3
CTAAACAACA
Seq_4
TTAAACATAG
Seq_5
GGACACATAG



	Seq_1	Seq_2	Seq_3	Seq_4	Seq_5
Seq_1	0	1	3	6	7
Seq_2	1	0	3	6	8
Seq_3	3	3	0	4	6
Seq_4	6	6	4	0	3
Seq_5	7	8	6	3	0

Figura 5.7 DNAs hipotéticos dos elementos analisados e quadro comparando suas diferenças dois a dois.

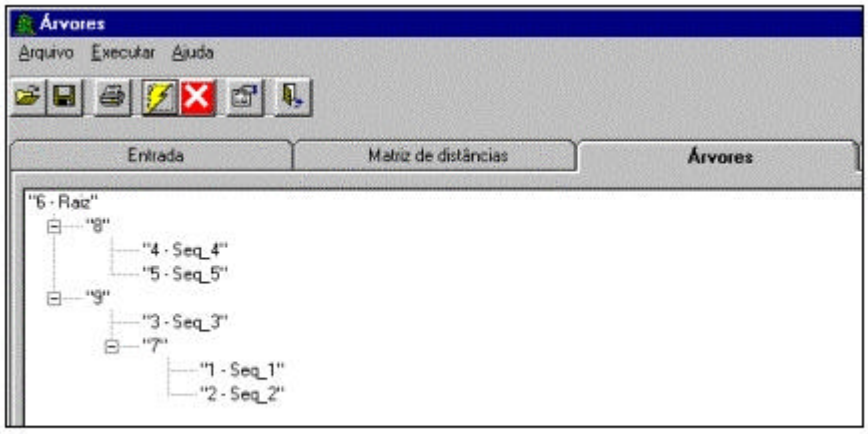


Figura 5.8 Árvore obtida utilizando o DNA descrito em 5.7 e processados apenas com o método da Matriz de Distâncias.

A Figura 5.8 mostra como ficou montada a árvore mais adequada para o conjunto de seqüências de DNAs estudado.

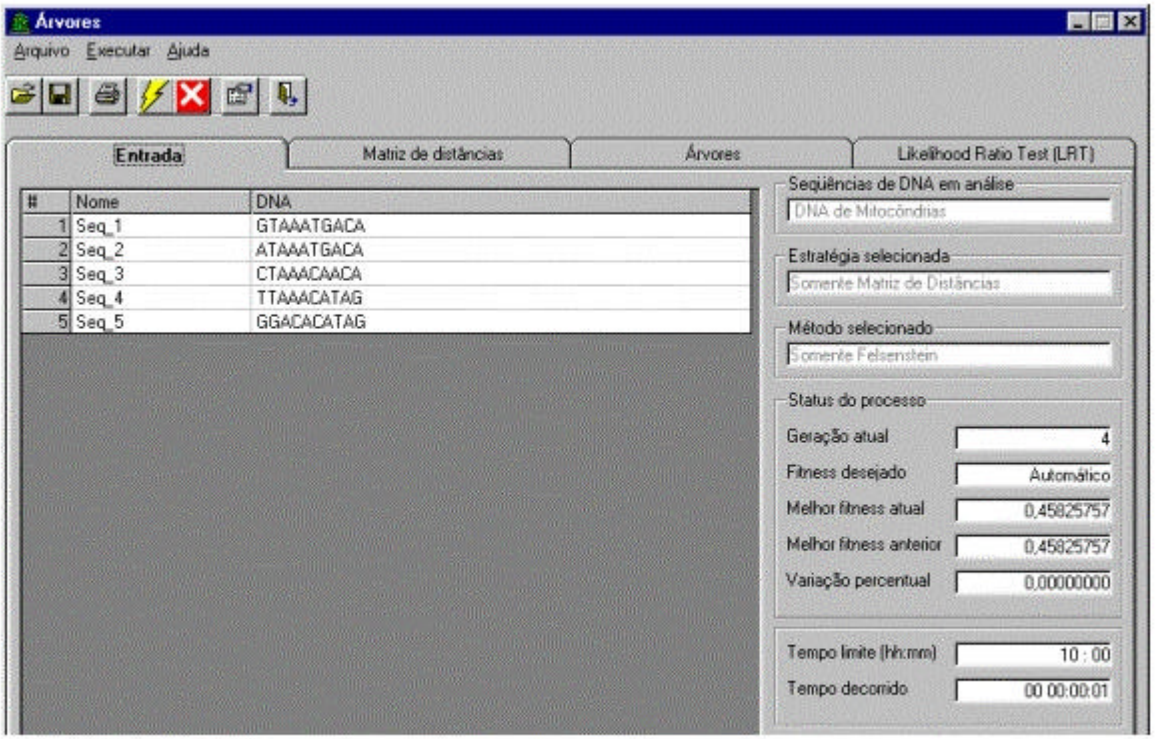


Figura 5.9 Tela principal da ferramenta computacional apresentando alguns dos principais parâmetros utilizados. Para uma descrição completa sobre os parâmetros deste software, veja Apêndice A.

Nas Figuras 5.10 a 5.12, serão exibidos os resultados obtidos referentes aos DNAs de dez seqüências de bases.

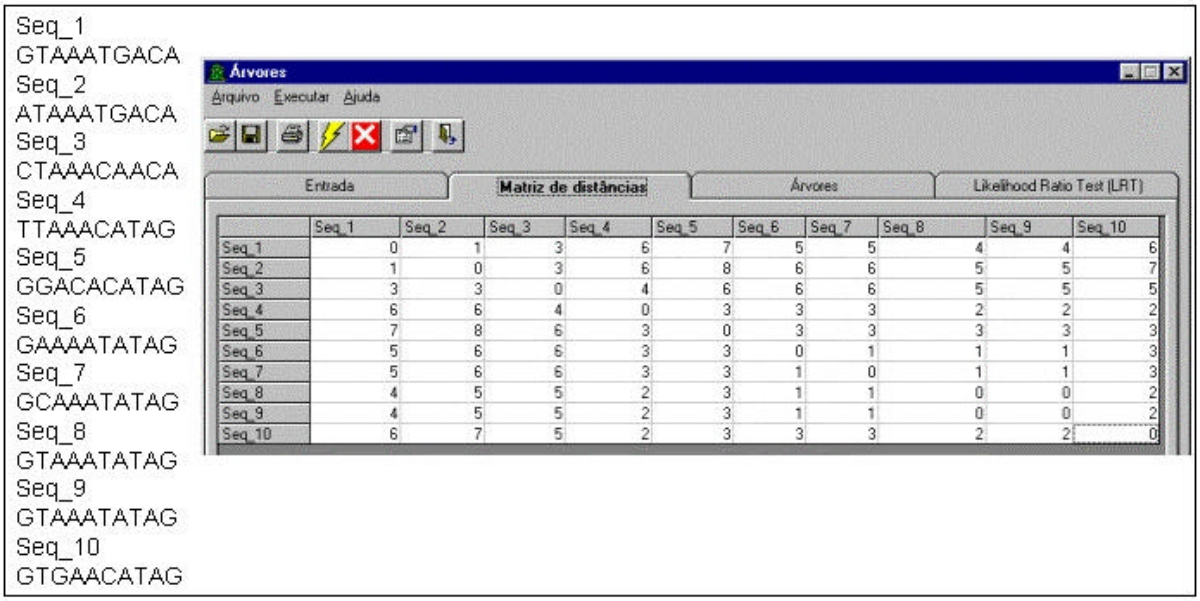


Figura 5.10 DNAs hipotéticos dos elementos analisados e quadro comparando suas diferenças dois a dois.

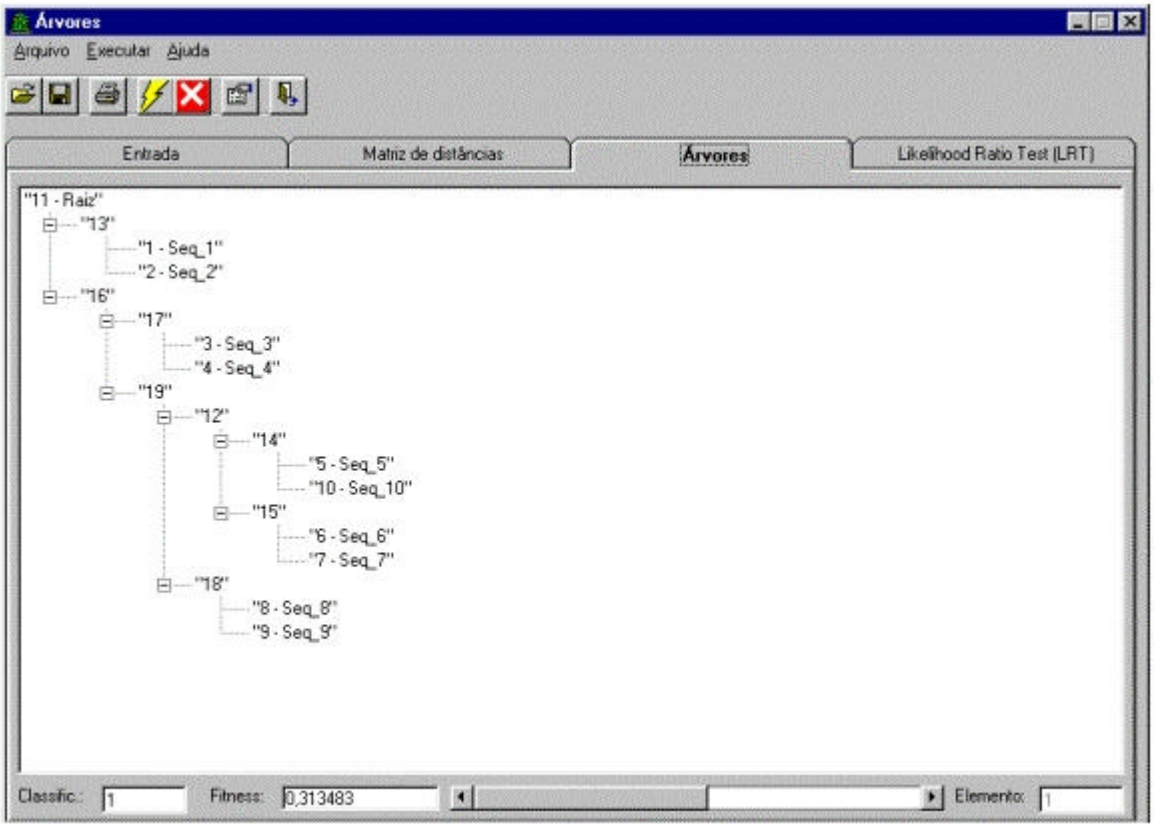


Figura 5.11 Árvore obtida utilizando o DNA descrito em 5.10 e processados apenas com o método da Matriz de Distâncias.

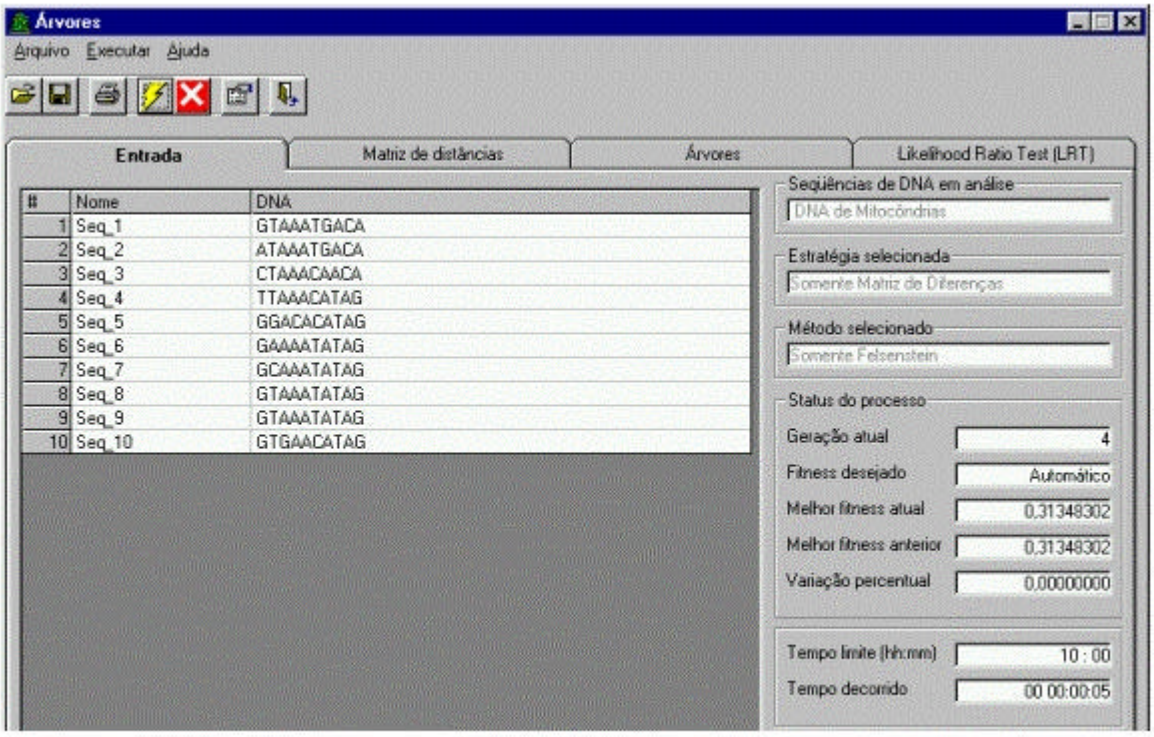


Figura 5.12 Tela principal da ferramenta computacional apresentando alguns dos principais parâmetros utilizados.

Nas Figuras 5.13 a 5.16 serão exibidos os resultados obtidos referentes aos DNAs de 20 seqüências hipotéticas de 40 bases cada.

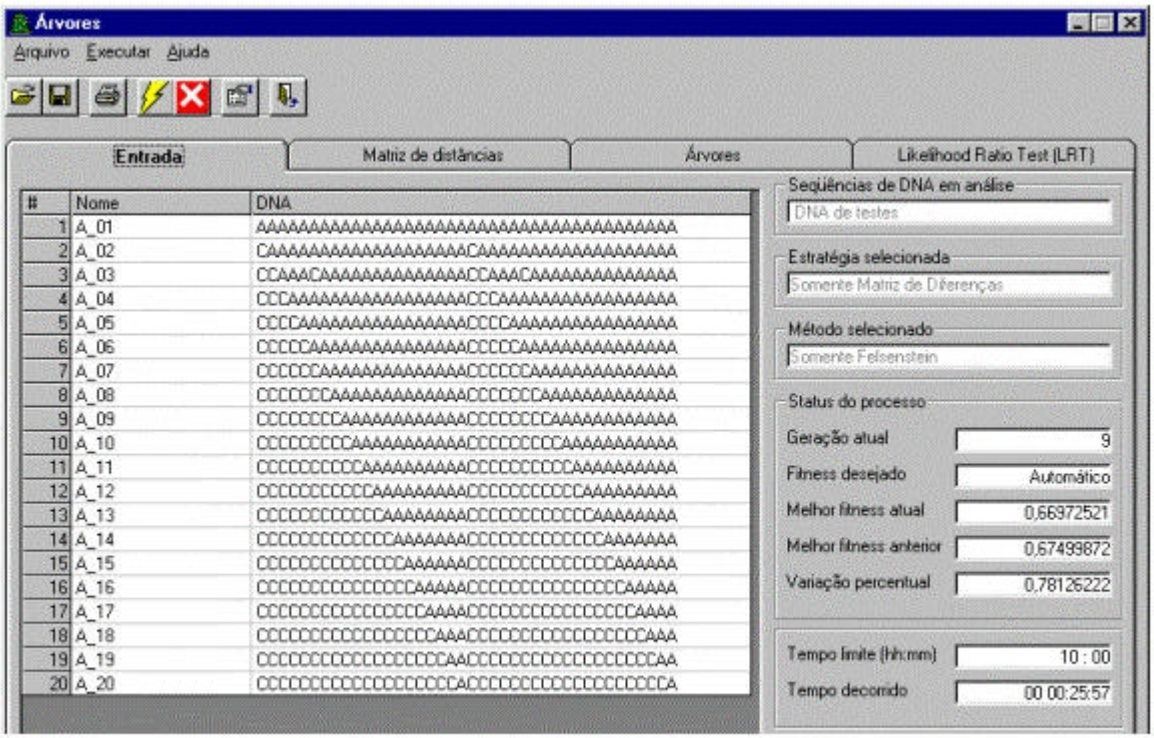
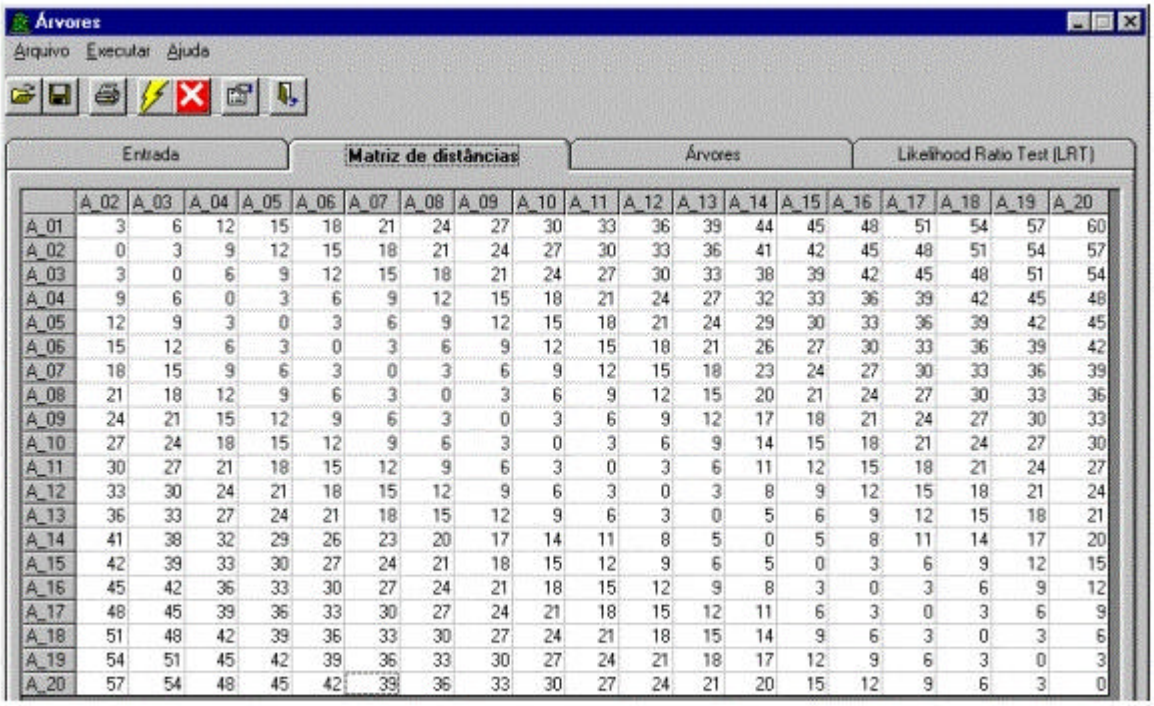


Figura 5.13 Tela principal da ferramenta computacional apresentando alguns dos principais parâmetros utilizados. Neste teste foram utilizadas 20 seqüências hipotéticas compostas por 40 bases cada.

Para tornar mais fácil a avaliação dos resultados, estas seqüências hipotéticas de bases foram especialmente criadas contendo diferenças variando de duas em duas unidades entre si, como pode ser observado na Figura 5.13, a seguir.



	A_02	A_03	A_04	A_05	A_06	A_07	A_08	A_09	A_10	A_11	A_12	A_13	A_14	A_15	A_16	A_17	A_18	A_19	A_20
A_01	3	6	12	15	18	21	24	27	30	33	36	39	44	45	48	51	54	57	60
A_02	0	3	9	12	15	18	21	24	27	30	33	36	41	42	45	48	51	54	57
A_03	3	0	6	9	12	15	18	21	24	27	30	33	38	39	42	45	48	51	54
A_04	9	6	0	3	6	9	12	15	18	21	24	27	32	33	36	39	42	45	48
A_05	12	9	3	0	3	6	9	12	15	18	21	24	29	30	33	36	39	42	45
A_06	15	12	6	3	0	3	6	9	12	15	18	21	26	27	30	33	36	39	42
A_07	18	15	9	6	3	0	3	6	9	12	15	18	23	24	27	30	33	36	39
A_08	21	18	12	9	6	3	0	3	6	9	12	15	20	21	24	27	30	33	36
A_09	24	21	15	12	9	6	3	0	3	6	9	12	17	18	21	24	27	30	33
A_10	27	24	18	15	12	9	6	3	0	3	6	9	14	15	18	21	24	27	30
A_11	30	27	21	18	15	12	9	6	3	0	3	6	11	12	15	18	21	24	27
A_12	33	30	24	21	18	15	12	9	6	3	0	3	8	9	12	15	18	21	24
A_13	36	33	27	24	21	18	15	12	9	6	3	0	5	6	9	12	15	18	21
A_14	41	38	32	29	26	23	20	17	14	11	8	5	0	5	8	11	14	17	20
A_15	42	39	33	30	27	24	21	18	15	12	9	6	5	0	3	6	9	12	15
A_16	45	42	36	33	30	27	24	21	18	15	12	9	8	3	0	3	6	9	12
A_17	48	45	39	36	33	30	27	24	21	18	15	12	11	6	3	0	3	6	9
A_18	51	48	42	39	36	33	30	27	24	21	18	15	14	9	6	3	0	3	6
A_19	54	51	45	42	39	36	33	30	27	24	21	18	17	12	9	6	3	0	3
A_20	57	54	48	45	42	39	36	33	30	27	24	21	20	15	12	9	6	3	0

Figura 5.14 Quadro comparando as diferenças entre as seqüências de bases utilizadas na Figura 5.13. Estas diferenças são apuradas comparando todos as seqüências duas a duas e o resultado obtido pode ser apresentado em formato matricial como o utilizado neste exemplo.

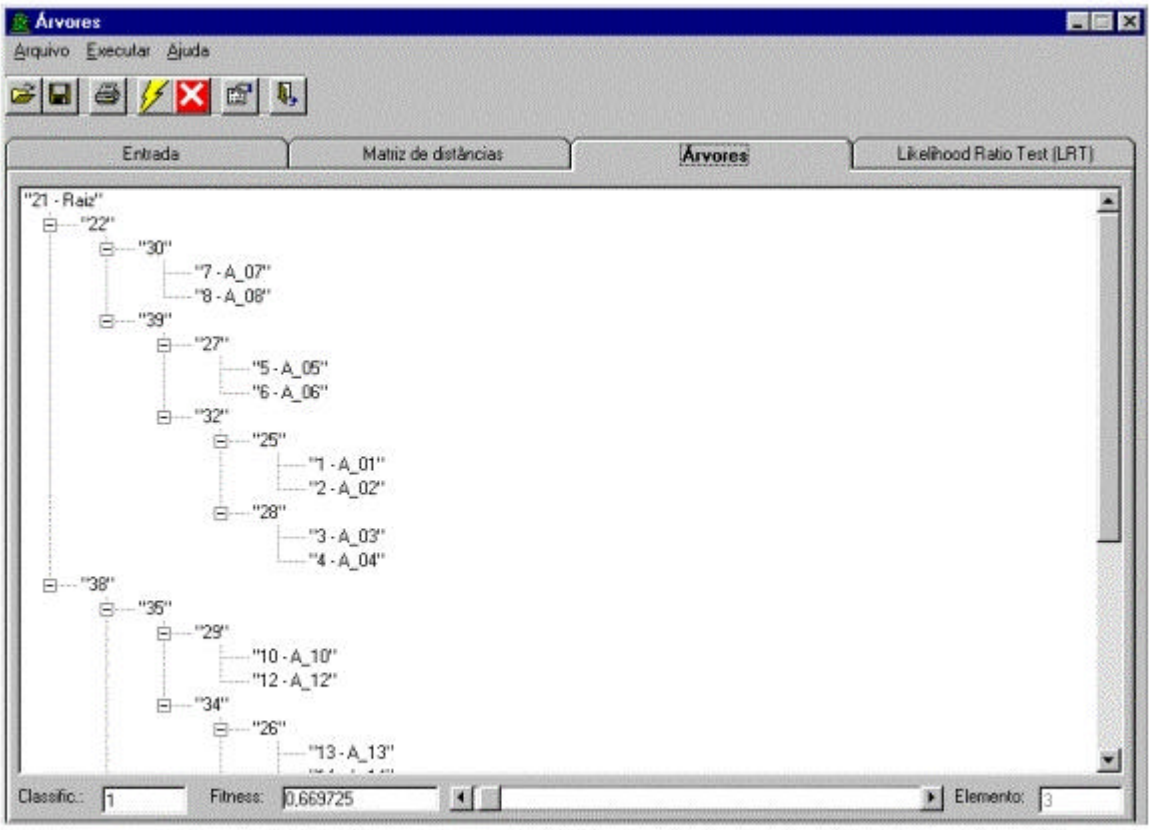


Figura 5.15 Metade superior da árvore obtida utilizando o DNA descrito em 5.13 e processados apenas com o método de Matriz de Distâncias.

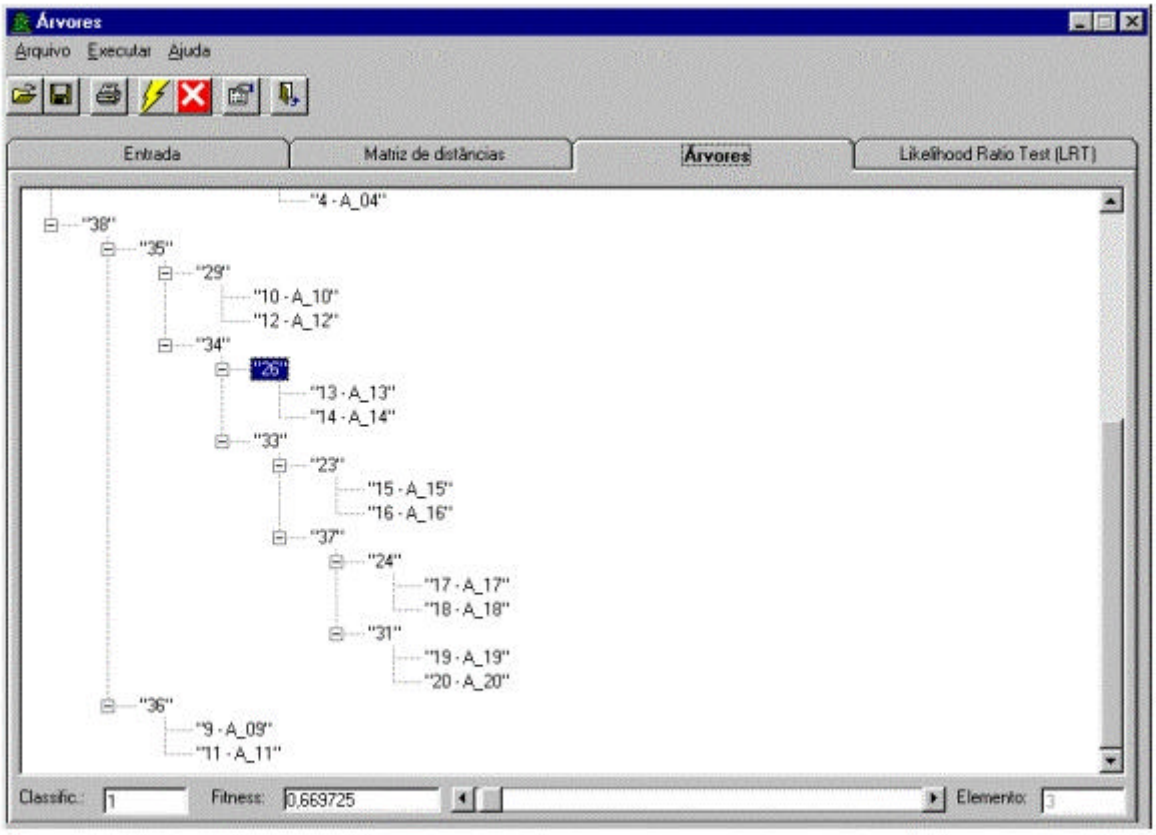


Figura 5.16 Metade inferior da árvore obtida utilizando o DNA descrito em 5.13 e processados apenas com o método de Matriz de Distâncias.

O nó em negrito é o ponto onde a árvore foi cortada para a divisão em duas partes. Na Figura 5.15 este nó se encontra na parte inferior da tela.

Nas Figuras 5.17 a 5.20, serão exibidos os resultados obtidos referentes aos DNAs de 50 seqüências compostas por 150 bases cada.

Como a recuperação de seqüências de bancos de genes tais como o GenBank foi muito difícil e também para tornar mais fácil a avaliação dos resultados, seqüências hipotéticas de bases foram especialmente criadas contendo diferenças variando de três em três unidades entre si, como pode ser observado na Figura 5.17, a seguir. Sem estas seqüências especialmente criadas não seria possível avaliar e ratificar os resultados das simulações. Como as seqüências apresentadas são bem conhecidas o resultado pode facilmente ser comprovado. Em todos os casos estudados o resultado obtido foi sempre o esperado, o que serviu para aumentar ainda mais nossa confiança nesta ferramenta computacional.

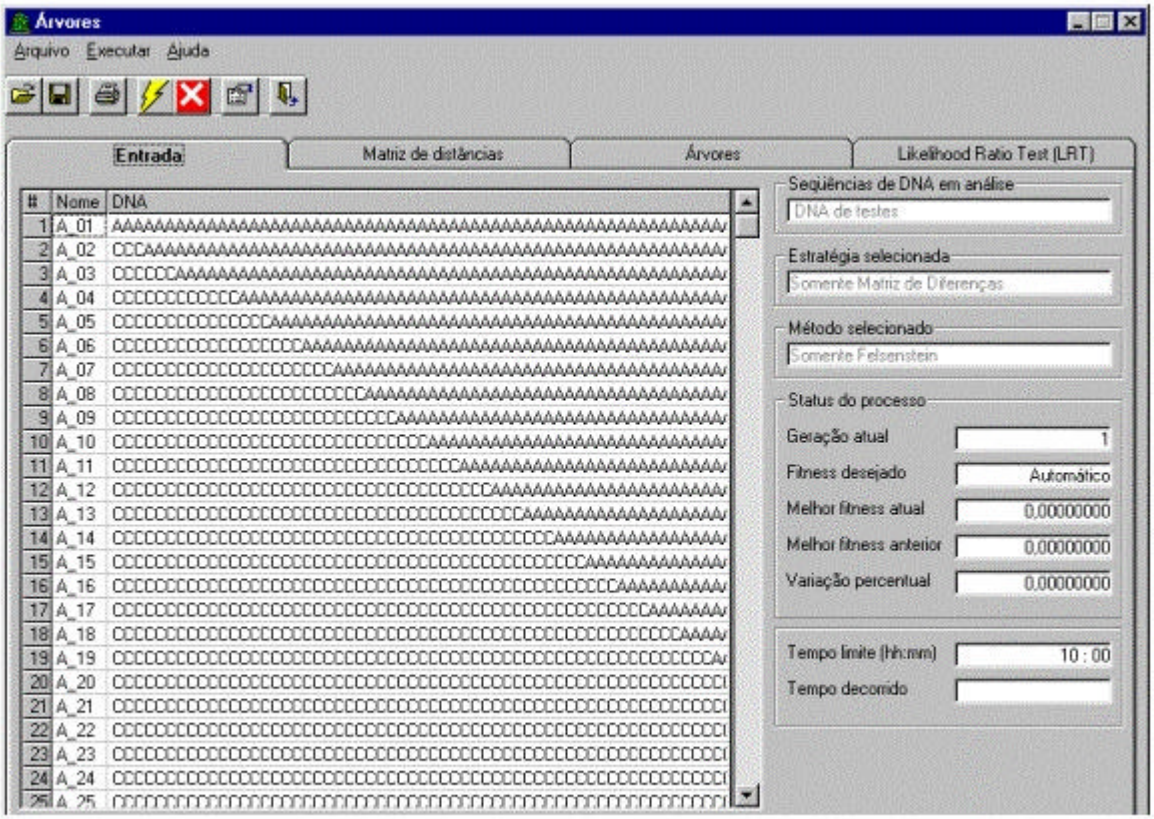


Figura 5.17 Tela principal da ferramenta computacional apresentando alguns dos principais parâmetros utilizados. Neste teste foram utilizadas 50 seqüências hipotéticas compostas por 150 bases cada.

Entrada | **Matriz de distâncias** | Árvores | Likelihood Ratio Test (LRT)

	A_01	A_02	A_03	A_04	A_05	A_06	A_07	A_08	A_09	A_10	A_11	A_12	A_13	A_14	A_15	A_16	A_17	A_18	A_19	A_20	A_21	A_
A_01	0	3	6	12	15	18	21	24	27	30	33	36	39	44	45	48	51	54	57	60	63	
A_02	3	0	3	9	12	15	18	21	24	27	30	33	36	41	42	45	48	51	54	57	60	
A_03	6	3	0	6	9	12	15	18	21	24	27	30	33	38	39	42	45	48	51	54	57	
A_04	12	9	6	0	3	6	9	12	15	18	21	24	27	32	33	36	39	42	45	48	51	
A_05	15	12	9	3	0	3	6	9	12	15	18	21	24	29	30	33	36	39	42	45	48	
A_06	18	15	12	6	3	0	3	6	9	12	15	18	21	26	27	30	33	36	39	42	45	
A_07	21	18	15	9	6	3	0	3	6	9	12	15	18	23	24	27	30	33	36	39	42	
A_08	24	21	18	12	9	6	3	0	3	6	9	12	15	20	21	24	27	30	33	36	39	
A_09	27	24	21	15	12	9	6	3	0	3	6	9	12	17	18	21	24	27	30	33	36	
A_10	30	27	24	18	15	12	9	6	3	0	3	6	9	14	15	18	21	24	27	30	33	
A_11	33	30	27	21	18	15	12	9	6	3	0	3	6	11	12	15	18	21	24	27	30	
A_12	36	33	30	24	21	18	15	12	9	6	3	0	3	8	9	12	15	18	21	24	27	
A_13	39	36	33	27	24	21	18	15	12	9	6	3	0	5	6	9	12	15	18	21	24	
A_14	44	41	38	32	29	26	23	20	17	14	11	8	5	0	5	8	11	14	17	20	23	
A_15	45	42	39	33	30	27	24	21	18	15	12	9	6	5	0	3	6	9	12	15	18	
A_16	48	45	42	36	33	30	27	24	21	18	15	12	9	8	3	0	3	6	9	12	15	
A_17	51	48	45	39	36	33	30	27	24	21	18	15	12	11	6	3	0	3	6	9	12	
A_18	54	51	48	42	39	36	33	30	27	24	21	18	15	14	9	6	3	0	3	6	9	
A_19	57	54	51	45	42	39	36	33	30	27	24	21	18	17	12	9	6	3	0	3	6	
A_20	60	57	54	48	45	42	39	36	33	30	27	24	21	20	15	12	9	6	3	0	3	
A_21	63	60	57	51	48	45	42	39	36	33	30	27	24	23	18	15	12	9	6	3	0	
A_22	66	63	60	54	51	48	45	42	39	36	33	30	27	26	21	18	15	12	9	6	3	
A_23	69	66	63	57	54	51	48	45	42	39	36	33	30	29	24	21	18	15	12	9	6	
A_24	72	69	66	60	57	54	51	48	45	42	39	36	33	32	27	24	21	18	15	12	9	

Figura 5.18 Quadro comparando as diferenças entre as seqüências de bases utilizadas na Figura 5.17. Estas diferenças são apuradas comparando todos as seqüências duas a duas e o resultado obtido pode ser apresentado em formato matricial como o utilizado neste exemplo.

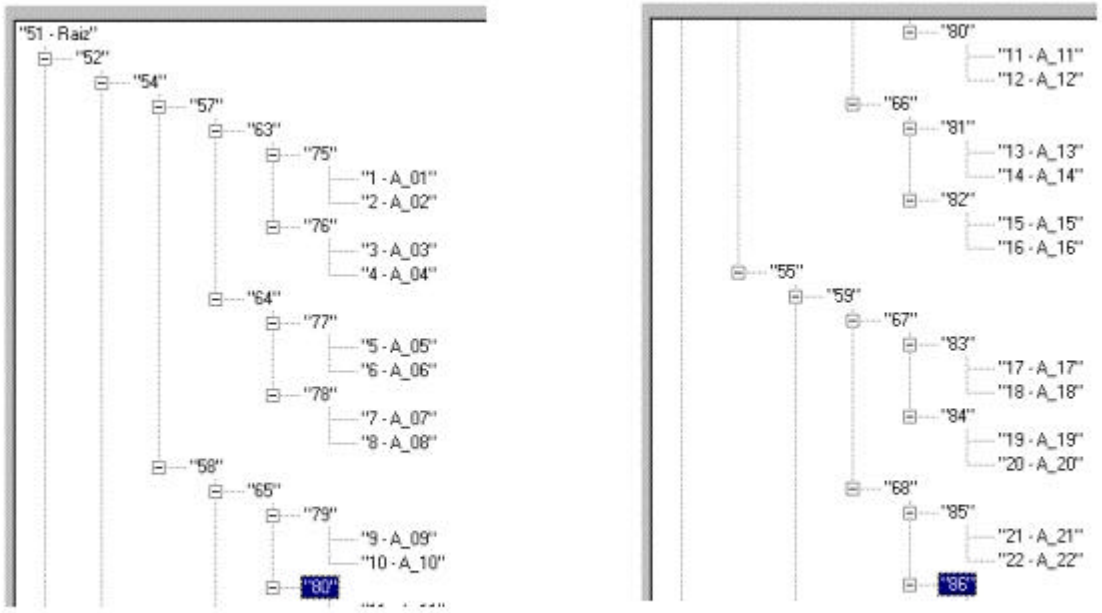


Figura 5.19 Metade superior da árvore obtida utilizando o DNA descrito em 5.17 e processados apenas com o método de Matriz de distâncias.

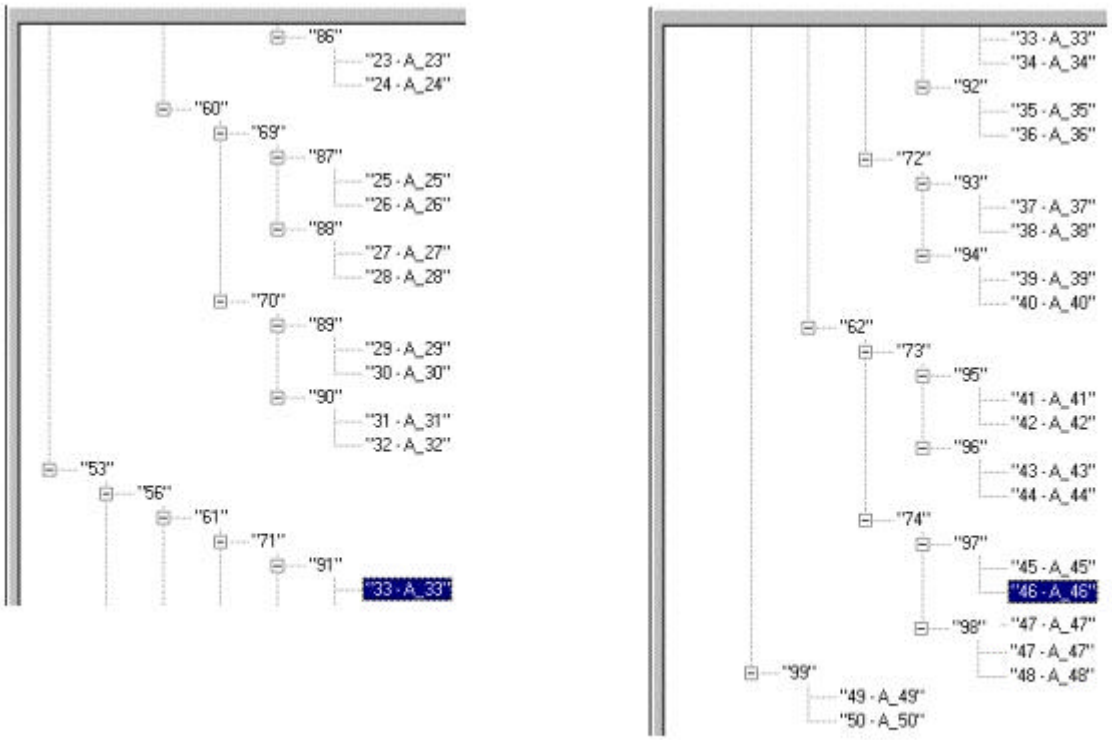


Figura 5.20 Metade inferior da árvore obtida utilizando o DNA descrito em 5.17 e processados apenas com o método de Matriz de Distâncias.

Os nós em negrito (26) são os pontos de referência onde a árvore foi cortada para a divisão nas quatro partes apresentadas nas Figuras 5.19 e 5.20.

Observações sobre o método da Máxima Verossimilhança (baseado em modelo)

A reconstrução de árvores filogenéticas usando o método da Máxima Verossimilhança requer que em algum momento sejam otimizados os comprimentos dos ramos das topologias analisadas. Em nossa pesquisa de mestrado foi usado um método de primeira ordem conhecido como “Método do Gradiente” para otimizar numericamente os comprimentos dos ramos. O “Método do Gradiente” foi aplicado conforme descrito por de Castro (1998) , onde se procura maximizar a superfície da máxima verossimilhança através dos ajustes realizados nos comprimentos dos ramos das árvores, efetuados de acordo com o sentido do gradiente.

A seguir, é apresentado o vetor gradiente G (informação de 1ª ordem) para um exemplo considerando uma árvore de quatro folhas:

$$G = \begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \end{bmatrix} = \begin{bmatrix} \frac{\partial \ln L}{\partial p_1} \\ \frac{\partial \ln L}{\partial p_2} \\ \frac{\partial \ln L}{\partial p_3} \\ \frac{\partial \ln L}{\partial p_4} \end{bmatrix} = \begin{bmatrix} \sum_j \frac{1}{L(j)} \cdot \frac{\partial L(j)}{\partial p_1} \\ \sum_j \frac{1}{L(j)} \cdot \frac{\partial L(j)}{\partial p_2} \\ \sum_j \frac{1}{L(j)} \cdot \frac{\partial L(j)}{\partial p_3} \\ \sum_j \frac{1}{L(j)} \cdot \frac{\partial L(j)}{\partial p_4} \end{bmatrix}$$, onde

$$\frac{\partial L(j)}{\partial p_1} = \sum_k \sum_l \pi_k \cdot P_{kl}(p_4) \cdot P_{ks_3}(p_3) \cdot \frac{\partial P_{ls_1}(p_1)}{\partial p_1} \cdot P_{ls_2}(p_2);$$

$$\frac{\partial L(j)}{\partial p_2} = \sum_k \sum_l \pi_k \cdot P_{kl}(p_4) \cdot P_{ks_3}(p_3) \cdot P_{ls_1}(p_1) \cdot \frac{\partial P_{ls_2}(p_2)}{\partial p_2};$$

$$\frac{\partial L(j)}{\partial p_3} = \sum_k \sum_l \pi_k \cdot P_{kl}(p_4) \cdot \frac{\partial P_{ks_3}(p_3)}{\partial p_3} \cdot P_{ls_1}(p_1) \cdot P_{ls_2}(p_2);$$

$$\frac{\partial L(j)}{\partial p_4} = \sum_k \sum_l \pi_k \cdot \frac{\partial P_{kl}(p_4)}{\partial p_4} \cdot P_{ks_3}(p_3) \cdot P_{ls_1}(p_1) \cdot P_{ls_2}(p_2);$$

Para $j = k$, $\frac{\partial P_{kk}(p_i)}{\partial p_i} = -1 + \pi_k$ e

Para $j \neq k$, $\frac{\partial P_{kj}(p_i)}{\partial p_i} = \pi_j$.

Algoritmo para método de 1ª ordem:

Para encontrar os valores dos comprimentos dos ramos das árvores, foi utilizado o Método do Gradiente (de Castro, 1998), conforme descrito a seguir:

- defina valores iniciais para o vetor *p*;
- defina um escalar ε > 0 arbitrariamente pequeno;
- tome 0 < *r* < 1 e *q* > 1; sugestão: $r = \frac{1}{2}$ e $q = 1.1$
- faça α = 1;
- calcule $\ln(L(j))$;
- calcule *G*;

- enquanto $\|G\| \geq \varepsilon$, faça:
    - $p^{provisório} = p + \frac{\alpha * G}{\|G\|}$;
    - calcule $\ln(L(j))^{provisório}$;
    - enquanto $\ln(L(j))^{provisório} \leq \ln(L(j))$ faça:
        - $\alpha = r\alpha$;
        - $p^{provisório} = p + \frac{\alpha * G}{\|G\|}$;
        - calcule $\ln(L(j))^{provisório}$;
    - fim
    - $p = p^{provisório}$;
    - $\ln(L(j)) = \ln(L(j))^{provisório}$
    - $\alpha = q\alpha$;
    - calcule *G*;
- fim

Este algoritmo ajusta os valores dos comprimentos dos ramos da árvore analisada de maneira a maximizar a "superfície de verossimilhança". Para uma ilustração gráfica do processo, vide a Figura 4.1 no Capítulo 4. A idéia básica consiste em realizar pequenos ajustes nos comprimentos dos ramos da árvore de acordo com o sentido do gradiente, de modo a maximizar a superfície de verossimilhança.

Conforme já descrito no Capítulo 3, devem ser calculadas as verossimilhanças de todas as árvores candidatas e a árvore que apresentar a máxima verossimilhança será considerada como sendo a árvore filogenética dos elementos analisados (Weir, 1996).

Nas figuras a seguir são exibidas algumas telas da ferramenta computacional, capturadas durante os processamentos de algumas seqüências de bases.

Na Figura 5.21 são apresentados os DNAs utilizados nos processamentos.

Humano
GTAAATATAGTTTAACCAAAACATCAGATTGTGAATCTGACAACAGAGGCTTACGACCCCTTATTTACC
Chimpanzé
GTAAATATAGTTTAACCAAAACATCAGATTGTGAATCTGACAACAGAGGCTCACGACCCCTTATTTACC
Gorila
GTAAATATAGTTTAACCAAAACATCAGATTGTGAATCTGATAACAGAGGCTCACAACCCCTTATTTACC
Orangotango
GTAAATATAGTTTAACCAAAACATTAGATTGTGAATCTAATAATAGGGCCCCACAACCCCTTATTTACC
Gibão
GTAAACATAGTTTAATCAAAACATTAGATTGTGAATCTAACAATAGAGGCTCGAAACCTCTTGCTTACC

Figura 5.21 DNA fornecido por Weir (1996).

Na Figura 5.22 são exibidos os resultados obtidos referentes aos DNAs das cinco seqüências de bases listadas na Figura 5.21 (Weir, 1996), usando o método da Máxima Verossimilhança com apoio de duas ferramentas diferentes. A primeira (a) foi obtida com PHYLIP de Felsenstein (Felsenstein, 1990; Weir, 1996) e o valor de sua máxima verossimilhança é -163,87052. A segunda (b) foi obtida com a ferramenta computacional desenvolvida ao longo desta pesquisa, chamada **Projeto Árvores Filogenéticas** e o valor de sua máxima verossimilhança é -171,11214.

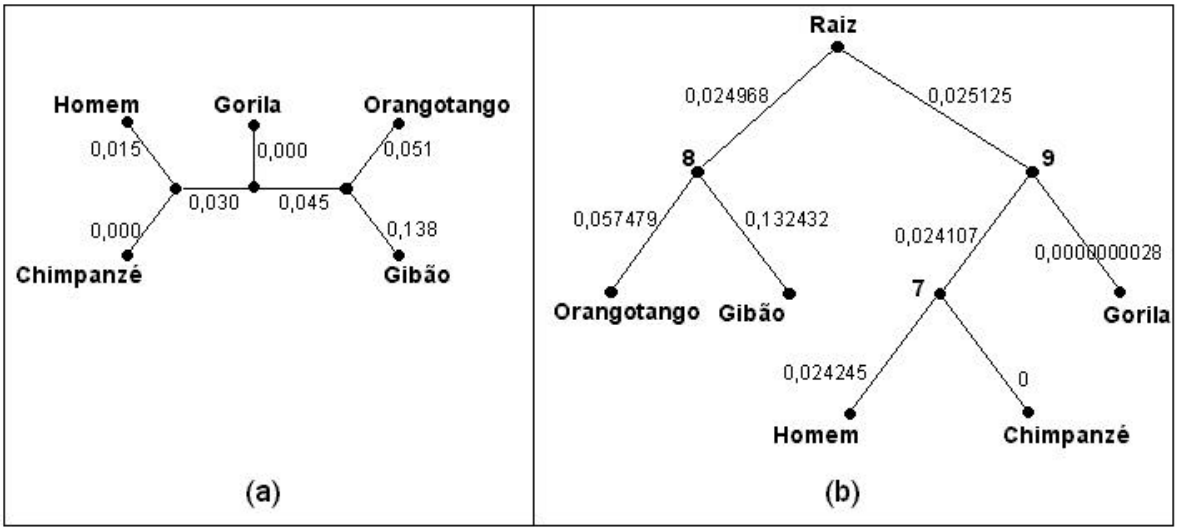


Figura 5.22 As árvores nesta figura representam o processamento do conjunto de DNAs da figura 5.21 usando duas ferramentas diferentes. Nos dois casos foram utilizados o método da Máxima Verossimilhança:
(a) É uma árvore do tipo estrela obtida como saída do programa PHYLIP de Felsenstein (Weir, 1996).
(b) É uma árvore com raiz obtida com a ferramenta computacional Projeto Árvores Filogenéticas.

Como a Figura 5.22 (a) apresenta uma árvore sem raiz e a Figura 5.22 (b) apresenta uma árvore com raiz não é possível comparar diretamente os resultados dos comprimentos dos ramos.

A tabela a seguir apresenta as distâncias somando os ramos de um elemento até outro.

Distâncias	Parte (a)	Parte (b)
Gorila – Gibão	0,183	0,1825
Gorila – Orangotango	0,096	0,1075
Gorila – Homem	0,045	0,0483
Gorila – Chimpanzé	0,030	0,0241
Gibão - Orangotango	0,189	0,1898
Gibão – Homem	0,228	0,2308
Gibão – Chimpanzé	0,213	0,2066
Homem - Chimpanzé	0,015	0,0242

Levando em consideração que as topologias das duas árvores são diferentes, as distâncias não são muito diferentes. Como a árvore sem raiz da parte (a) possui uma quantidade menor de ramos para serem otimizados, não surpreende o fato de sua máxima verossimilhança ser maior que o da parte (b).

A Figura 5.23 apresenta a tela do **Projeto Árvores Filogenéticas** com o resultado do processamento para estas seqüências de DNAs.

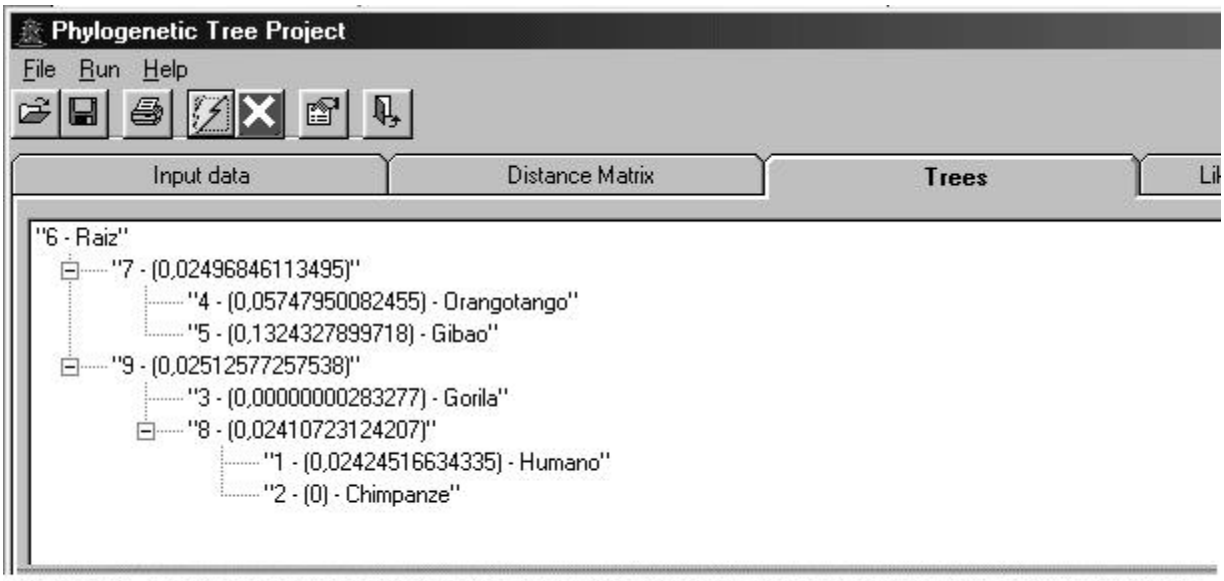


Figura 5.23 Resultado do processamento das seqüências apresentadas do quadro da Figura 5.21 Weir (1996). Para este processamento foi utilizado o método da Máxima Verossimilhança e o modelo de substituição de bases de Felsenstein.

Nas Figuras 5.24 e 5.25 são exibidos mais alguns dos resultados obtidos referentes aos DNAs de quatro seqüências, retiradas do quadro da Figura 5.21, usando somente o método da

Máxima Verossimilhança e os modelos de substituição de bases de Jukes-Cantor (1969), Felsenstein (1981) e Kimura (1980).

A Figura 5.24 apresenta os resultados obtidos ao se calcular os comprimentos dos ramos da árvore montada com estas quatro seqüências de bases. Nesta simulação, o modelo de substituição de bases de Felsenstein (1981) mostrou-se bem melhor que os outros dois. A comparação entre os resultados foi realizada utilizando a fórmula *LRT* descrita mais adiante neste capítulo. Esta fórmula é utilizada para comparar os resultados obtidos por modelos diferentes. Neste caso, ela foi utilizada para comparar os resultados obtidos pelos três modelos de substituição de bases pesquisados (Jukes-Cantor, 1969; Felsenstein, 1981 e Kimura, 1980).

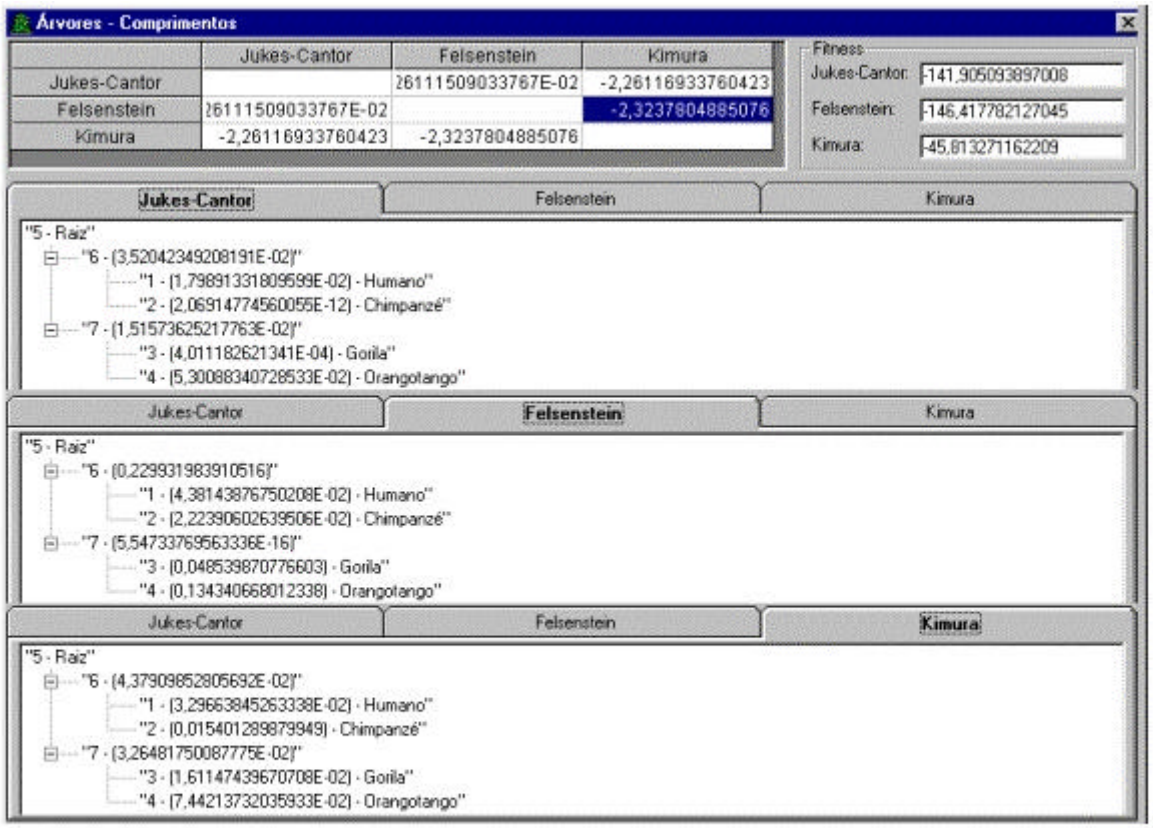


Figura 5.24 Árvores obtidas utilizando DNA fornecido por Weir (1996) e apresentado na Figura 5.21. Para este processamento, foi utilizado somente o método da Máxima Verossimilhança comparando os resultados obtidos com o uso dos modelos de substituição de bases de Jukes-Cantor, Kimura e Felsenstein.

As diferenças encontradas para os comprimentos dos ramos das três árvores são esperadas porque cada uma delas foi calculada seguindo um modelo de substituição de bases

diferente. Estes modelos já foram comentados nos capítulos anteriores. O quadro no alto da Figura 5.24 foi calculado usando a fórmula "*Likelihood Ratio Test"* (Wu & Li, 1985):

$$LRT = -2 * \ln\left(\frac{L_a}{L_b}\right)$$

Esta fórmula é utilizada para comparar os resultados do emprego de modelos diferentes para construção de árvores filogenéticas. Neste exemplo o melhor resultado foi obtido com o emprego do modelo de Felsenstein (1981) em relação a Kimura (1980). Esta fórmula sempre compara um modelo em relação a outro. Caso existam mais de dois modelos em análise, eles devem ser medidos e comparados dois a dois. Maiores esclarecimentos sobre este conceito podem ser obtidos em Muse & Weir (1992).

A Figura 5.25 apresenta uma parte das seqüências utilizadas e alguns dos parâmetros usados nesta busca. Não foi possível mostrar as seqüências inteiras por causa de seu tamanho bem superior ao limite da tela. Também são mostrados mais alguns dados interessantes sobre o conjunto de seqüências pesquisado, tais como o nome dado ao conjunto pelo pesquisador que é exibido no campo "Seqüências de DNA em análise" e neste caso contém o nome "DNA de Mitocôndrias (Weir, 1996)". Logo abaixo deste campo pode ser encontrado o campo onde é informada a estratégia selecionada para a busca (Matriz de Distâncias, Máxima Verossimilhança ou uma composição das duas que foi chamada de Estratégia Mista), e logo abaixo se encontra o campo onde é informado o modelo de substituição de bases utilizado. Estão disponíveis no momento os modelos de Jukes-Cantor (1969), Felsenstein (1981) e Kimura (1980);. Neste caso foi selecionado que, após o final da busca pela topologia mais adequada, a ferramenta apresente os resultados comparativos obtidos pelos três modelos (Método comparativo).

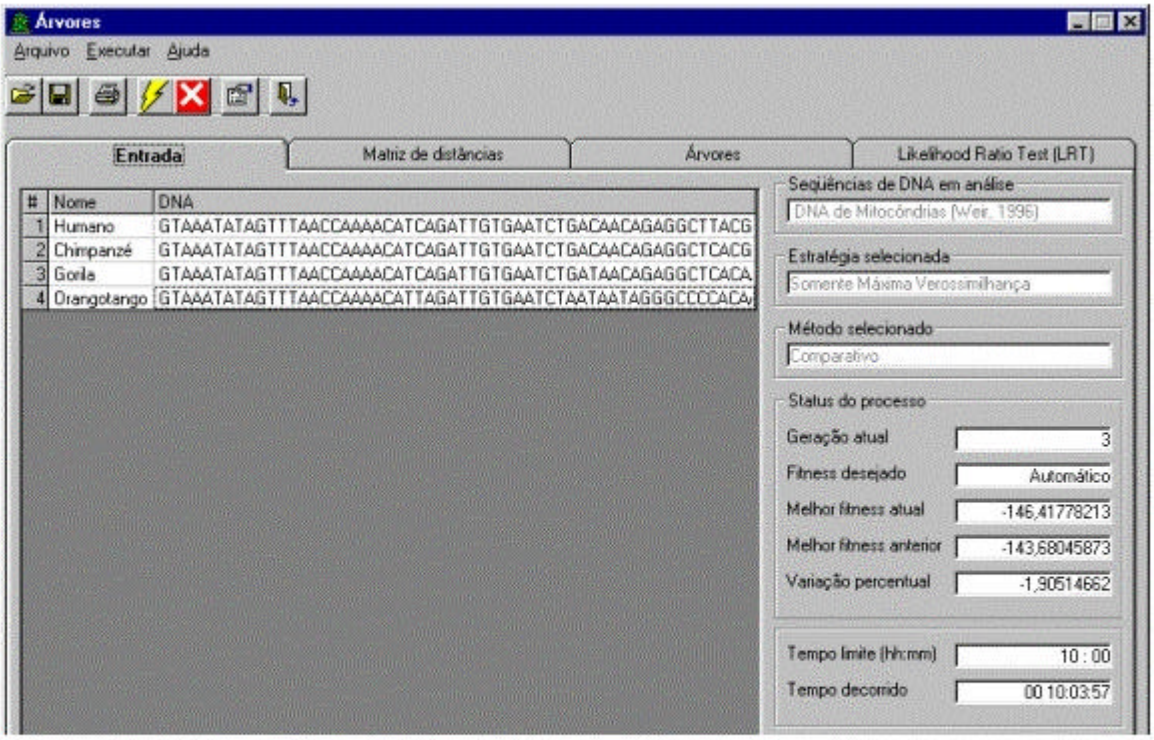


Figura 5.25 Tela principal do Projeto Árvores Filogenéticas, exibindo os DNAs utilizados, o tempo decorrido durante a busca e mais alguns parâmetros estabelecidos pelo usuário. A estratégia empregada foi somente a Máxima Verossimilhança utilizando os modelos de substituição de bases de Jukes-Cantor, Felsenstein e Kimura. Maiores detalhes sobre os parâmetros do software podem ser encontrados no Apêndice A.

Nas Figuras 5.26 e 5.27 são exibidos os resultados obtidos referentes aos mesmos DNAs de quatro espécies, mas usando agora a estratégia mista: iniciar processamento usando o método da Matriz de Distâncias e finalizar calculando os comprimentos dos ramos da melhor árvore encontrada, utilizando o método da Máxima Verossimilhança.

A Figura 5.26 apresenta os resultados obtidos usando a estratégia mista (início da busca com Matriz de Distâncias e finalização da busca com Máxima Verossimilhança) e uma comparação dos resultados obtidos utilizando os três modelos de substituição de bases disponíveis.

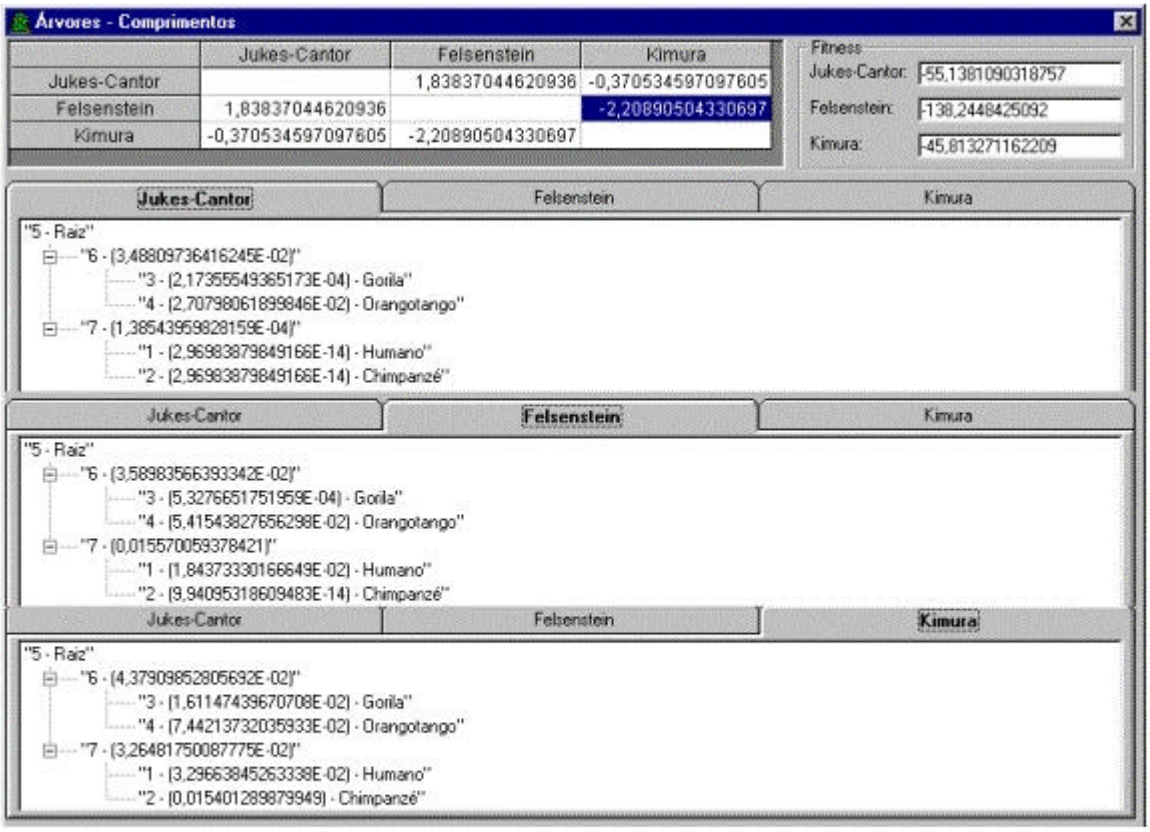


Figura 5.26 Árvores obtidas utilizando DNA de Weir (1996) e apresentado na Figura 5.21. Primeiro, os dados foram processados com Matriz de Distâncias, até encontrar uma árvore candidata, que depois teve os comprimentos de seus ramos calculados pelo método da Máxima Verossimilhança. Também estão sendo exibidos os resultados obtidos com o uso dos modelos de substituição de bases de Jukes-Cantor, Kimura e Felsenstein.

Nesta simulação o modelo de substituição de bases de Felsenstein (1981) também obteve o melhor resultado usando a fórmula *LRT*, já comentada anteriormente. Este resultado comparado ao da Figura 5.24 serve para ressaltar a superioridade do método da Máxima Verossimilhança em relação a outros, mesmo em relação a este método alternativo que inicia a busca com Matriz de Distâncias e termina com a determinação dos comprimentos dos ramos usando as técnicas da Máxima Verossimilhança, descritas por Weir (1996) .

O software utilizado para estas simulações (Projeto Árvores Filogenéticas) permite que o pesquisador decida qual método deverá ser utilizado para a análise das seqüências de bases (Matriz de Distâncias ou Máxima Verossimilhança ou uma combinação de ambos) e permite também combinar as opções anteriores com um dos três modelos de substituição de bases estudados ao longo deste trabalho de mestrado (Jukes-Cantor, 1969; Felsenstein, 1981 e Kimura, 1980). Além destes parâmetros específicos sobre filogenia, este software permite

também ajustar os parâmetros específicos do algoritmo genético, tais como, porcentagem de mutações, tamanho da população, quantidade de iterações e tempo máximo de busca. Como estes parâmetros podem não ser bem compreendidos por todos os pesquisadores que vierem a utilizar este software, foi fixado um conjunto de valores iniciais (*default*) que podem ser alterados sempre que necessário. A intenção em fornecer estes valores iniciais foi somente facilitar o uso desta ferramenta e evitar que todos os parâmetros devam sempre ser digitados antes de cada execução. Estes valores foram otimizados ao longo desta pesquisa de mestrado e se mostraram muito bons na maioria dos casos analisados.

A Figura 5.27 apresenta basicamente as mesmas informações que a Figura 5.25, pois foram utilizadas as mesmas seqüências de bases, apenas alterando a estratégia de busca. Neste teste foi utilizada a estratégia mista, que inicia a busca com o método Matriz de Distâncias e finaliza com Máxima Verossimilhança.

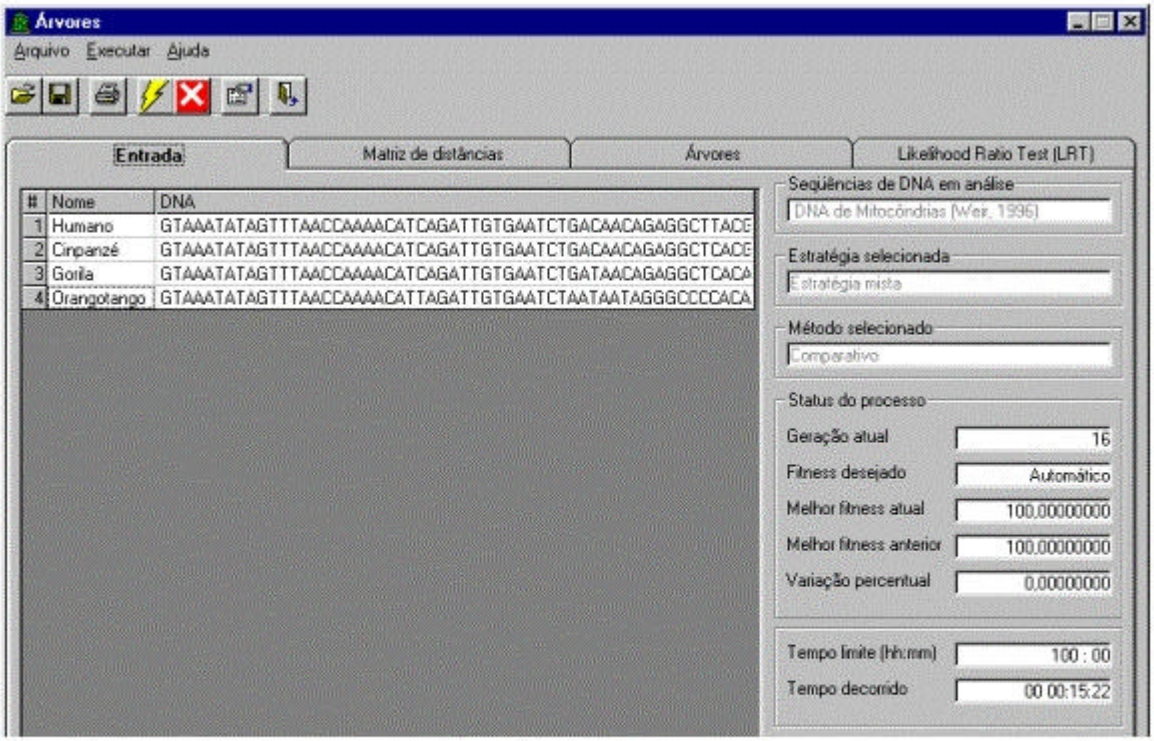


Figura 5.27 Tela principal do Projeto Árvores Filogenéticas, exibindo os DNAs utilizados, o tempo decorrido durante a busca e mais alguns parâmetros estabelecidos pelo usuário. Estratégia mista significa iniciar a busca usando o método da Matriz de Distâncias e finalizar a busca calculando os comprimentos dos ramos com o método da Máxima Verossimilhança, utilizando algum modelo de substituição de bases (Jukes-Cantor, Felsenstein ou Kimura). Maiores detalhes sobre os parâmetros do software podem ser encontrados no Apêndice A.

Conclusões

Todos os resultados apresentados neste trabalho de mestrado foram obtidos com o uso de uma única ferramenta computacional. Este software (Projeto Árvores Filogenéticas) foi desenvolvido e utilizado ao longo desta pesquisa de mestrado e oferece ao pesquisador grande variedade de opções para trabalhar. É possível alterar os parâmetros sobre filogenia, tais como estratégia de busca (Matriz de Distâncias ou Máxima Verossimilhança) e modelo de transição de bases (Jukes-Cantor, 1969; Felsenstein, 1981 e Kimura, 1980) e também é possível alterar os parâmetros sobre o algoritmo genético utilizado internamente. Não é necessária a digitação de todos estes parâmetros, pois a ferramenta possui um conjunto de valores iniciais que se mostraram bastante eficientes na maioria dos testes realizados.

A interface gráfica deste software foi projetada para facilitar seu uso por parte do usuário, mesmo que este não domine completamente as técnicas de computação. No Apêndice A, pode ser encontrado um exemplo completo de seu uso, passo a passo, onde são explicados detalhadamente todos os campos de suas telas e todas as opções disponíveis para sua execução. É necessário ressaltar que este software foi projetado para auxiliar e orientar o usuário durante o processo de execução. Sempre que um dado for necessário ou um parâmetro for indispensável e não estiver nos valores iniciais, o usuário será solicitado a fornecer este dado ou informação faltante. Por exemplo, o nome do arquivo que contém o conjunto de seqüências a serem analisadas não pode fazer parte do conjunto de valores iniciais, razão pela qual ele é solicitado sempre que o usuário ordenar o início de execução. Uma tela específica de busca de arquivos no padrão Windows será exibida para que o usuário aponte para o arquivo desejado.

Os recursos e as facilidades disponíveis nesta ferramenta ajudaram muito o desenvolvimento deste trabalho de mestrado e poderão ser úteis também para novas pesquisas sobre Filogenia e sobre Computação Evolutiva.

Capítulo 6

Conclusões e Perspectivas Futuras

6.1 Comentários gerais sobre o trabalho

Ao longo desta pesquisa de mestrado, foi possível estudar, analisar e experimentar algumas técnicas e conceitos muito úteis e interessantes voltados para filogenia. Foram explorados basicamente os métodos da Matriz de Distâncias e da Máxima Verossimilhança, além dos modelos de substituição de bases de Jukes-Cantor (1969), Felsenstein (1981) e Kimura (1980), que são os mais comumente encontrados na literatura especializada.

O desejo de reconstruir uma árvore para representar, de maneira hierárquica (ancestrais e descendentes), todos os organismos do planeta é bem antigo, sendo que o primeiro diagrama conhecido sobre este assunto é o que Lamark criou em 1809. Darwin usou um diagrama deste tipo em seu livro “Origem das Espécies” (1859), mas foi somente em 1866 que E. Haeckel apresentou um diagrama em forma de árvore abrangendo todos os grupos de seres vivos. Sua árvore filogenética foi concebida com base em aspectos muito questionáveis do ponto de vista científico e não pode ser comparada tecnicamente às que modernamente se organizam (Enciclopédia Mirador Internacional, 1995).

Os métodos mais aceitos para estudo sobre filogenia se apóiam no modelo evolucionário, como já foi explicado em capítulos anteriores (Nei & Kumar, 2000). Espera-se que a filogenia molecular venha a esclarecer muitos padrões de ramos da árvore da vida que têm sido difíceis de resolver com métodos baseados em características morfológicas e fisiológicas. Não se pode ignorar, no entanto, a existência de uma dificuldade intrínseca: dada uma seqüência de nucleotídeos, não se constitui uma tarefa simples a atribuição de grau de funcionalidade a cada elemento da seqüência, que conduziria a uma análise filogenética ponderada. O que se faz geralmente, e também foi adotado nesta pesquisa, é atribuir o mesmo grau de funcionalidade a todos os elementos da seqüência de nucleotídeos.

O que se conclui, portanto, é que todos os métodos já propostos para realizar análise filogenética (baseados em Filogenia, Morfologia e Fisiologia) vão subsistir e devem ser

considerados conjuntamente, sempre que possível. As vantagens e desvantagens de cada método devem ser devidamente investigadas, principalmente na ausência de consenso entre as respostas fornecidas para um dado conjunto de dados de entrada.

Baseado no modelo evolucionário, é possível reconstruir uma árvore filogenética a partir de um conjunto de seqüências de bases de algumas espécies. Existem diversos métodos para realizar esta reconstrução, sendo que eles podem ser divididos em dois tipos fundamentais:

- Não baseados em modelo, representado pelo método da Matriz de Distâncias, que apenas leva em consideração as distâncias apuradas entre as seqüências de bases analisadas;
- Baseados em modelo, representado pelo método da Máxima Verossimilhança. Este método faz uso explícito de todas as informações disponíveis nas seqüências de bases analisadas, com apoio de algum modelo probabilístico, e não se restringe somente às distâncias entre os pares de bases (Cavalli-Sforza & Edwards, 1967).

Os detalhes sobre cada um destes dois métodos, bem como suas vantagens e desvantagens, além de alguns outros métodos, foram amplamente discutidos nos Capítulos 3 e 5.

Para auxiliar o estudo e também para permitir realizar experimentos sobre alguns dos métodos disponíveis, foi desenvolvida uma ferramenta computacional denominada “Projeto Árvores Filogenéticas”, que apresenta as seguintes características:

- Sua interface gráfica é amigável e foi projetada para facilitar seu uso por parte de usuários, mesmo que estes não dominem completamente as técnicas de computação;
- Permite ajuste dos principais parâmetros relativos à filogenia, tais como seleção do método de busca a ser utilizado (Matriz de Distâncias ou Máxima Verossimilhança) e modelo de substituição de bases (Jukes-Cantor, 1969; Felsenstein, 1981 ou Kimura, 1980);
- Permite ajuste dos principais parâmetros relativos ao algoritmo genético utilizado, tais como tamanho da população e porcentagem de mutações.

Os detalhes sobre este software e as orientações sobre sua utilização encontram-se no Apêndice A.

As seguintes características não foram implementadas nesta versão do “Projeto Árvores Filogenéticas”:

- Possibilidade de escolher algum outro método de busca diferente dos dois estudados (Matriz de Distâncias e Máxima Verossimilhança);
- Possibilidade de escolher algum outro modelo de substituição de bases diferente dos três disponíveis (Jukes-Cantor, 1969; Felsenstein, 1981 ou Kimura, 1980);
- Possibilidade de execução do software em algum outro sistema operacional diferente de MS Windows.

Apoiado por este software, foi possível constatar o poder da Computação Evolutiva para resolver este complexo problema, cujo espaço de solução cresce com velocidade fatorial em função do número de seqüências de bases analisadas.

Problemas complexos como este impedem o uso de técnicas convencionais para sua solução. É praticamente impossível analisar todas as árvores candidatas (busca exaustiva) quando o número de seqüências de bases for superior a umas poucas dezenas (Lewis, 1998). Esta foi a principal motivação que levou ao uso de Computação Evolutiva nesta pesquisa de mestrado. As técnicas de Computação Evolutiva permitem explorar e explotar de maneira muito eficiente o espaço de solução e, geralmente, uma boa resposta pode ser encontrada em tempo razoável (Bäck *et al*., 1997). É preciso ressaltar que não existem garantias de ser encontrada a melhor solução para o problema: o que se garante é uma boa solução, caso se proceda a uma escolha adequada de parâmetros estabelecidos para a busca. As técnicas relativas à Computação Evolutiva foram discutidas no Capítulo 2.

O primeiro trabalho publicado sobre o uso de Computação Evolutiva para reconstrução de árvores filogenéticas foi o de Matsuda (1995) . Existem mais algumas publicações sobre este assunto, como pode ser verificado ao longo do Capítulo 4. A principal diferença entre os trabalhos analisados e esta dissertação de mestrado é a maneira como o assunto foi abordado. Enquanto as outras obras tratam o problema de maneira polarizada, se concentrando fortemente nos aspectos relacionados à Biologia e muito pouco nos aspectos relacionados à Engenharia, neste trabalho se buscou tratar o problema de maneira holística, detalhando e explicando todos os passos necessários para a reconstrução de uma boa árvore filogenética, a partir de uma seqüência de bases, sem deixar de levar em consideração todos os aspectos mais relevantes em relação à Biologia e à Engenharia. Foram esclarecidos todos os passos necessários para esta busca: desde o modelo evolucionário, passando pelos métodos de busca disponíveis, comentando os modelos de substituição de bases normalmente empregados, até chegar ao método do gradiente (método de otimização de primeira ordem), que foi utilizado

para o ajuste dos comprimentos dos ramos das árvores. Em nenhum dos trabalhos analisados foi encontrada uma descrição completa de todos os passos necessários. Foi preciso consultar diversas obras para poder chegar ao entendimento deste processo inteiro.

Os detalhes sobre o algoritmo genético utilizado e sobre o método de otimização de primeira ordem utilizado (gradiente) podem ser encontrados no Capítulo 5. Neste capítulo, também foram exibidos alguns dos resultados da pesquisa.

6.2 Conclusões

Apesar do método da Máxima Verossimilhança ter sido proposto em 1967 (Cavalli-Sforza & Edwards, 1967), somente em meados da década de 1990 ele se tornou viável, em virtude de seu elevado custo computacional. Durante muito tempo os métodos baseados em Matriz de Distâncias foram muito utilizados por sua facilidade de implementação e reduzido custo computacional (Nei & Kumar, 2000).

Os dois métodos (Matriz de Distâncias e Máxima Verossimilhança) são úteis e eficientes dentro de seus limites de ação, potencialidade e finalidade. Cada um deles pode ser aplicado com sucesso, desde que se tenha em mente o que se deseja obter.

Caso o interesse seja apenas o de encontrar uma árvore filogenética que represente as seqüências de bases analisadas, sem levar em consideração os comprimentos dos ramos, e que represente os graus de parentesco, sem se preocupar com qual ramo é mais comprido ou mais curto do que outro, pode-se usar o método da Matriz de Distâncias, que é bem mais econômico em termos de processamento. A busca é mais rápida e a árvore encontrada geralmente possui uma topologia próxima, quando não coincidente, àquela encontrada usando Máxima Verossimilhança.

Caso o interesse seja estudar as distâncias entre os elementos, além de seus graus de parentesco, então o método da Máxima Verossimilhança deve ser empregado, mesmo sendo mais caro em termos de processamento, pois somente com ele é possível calcular os comprimentos dos ramos da árvore encontrada.

Filogenia não é um assunto recente, desde os tempos de Darwin se procura montar a árvore das espécies. Como o espaço de soluções deste problema é gigantesco (apresenta custo fatorial), o emprego de algoritmos genéticos se apresenta como sendo uma boa alternativa para sua exploração. Este espaço de solução atinge a ordem de 10^{76} para apenas 50 seqüências de bases analisadas. O uso de Computação Evolutiva se mostrou muito eficiente

e eficaz em sua exploração ao longo desta pesquisa. Em todas as simulações, sempre foi obtido um bom resultado dentro de um período de tempo razoável.

Os aspectos relacionados com o ajuste dos comprimentos dos ramos foram os menos esclarecidos em todas as obras consultadas para a realização desta pesquisa. A maioria dos autores apenas informa que estes comprimentos podem ser obtidos com apoio de algum método numérico mas nenhum deles chega a detalhar uma solução completa do início ao fim. Neste trabalho, ao longo do Capítulo 5, foram apresentados em detalhes todos os passos necessários para o ajuste dos comprimentos dos ramos, baseado no método de primeira ordem conhecido como método do gradiente.

Ainda existe grande carência de ferramentas computacionais à disposição dos pesquisadores desta área (filogenia) e as atualmente disponíveis apresentam interface gráfica pouco amigável. A ferramenta computacional desenvolvida ao longo desta pesquisa de mestrado procurou simplificar ao máximo a interface gráfica com o usuário de modo a viabilizar seu uso por profissionais que não dominem as técnicas de computação. A maioria de seus parâmetros não precisa ser alterada e já é inicializada de maneira automática quando o software é executado. Contudo, estes parâmetros podem ser alterados caso o pesquisador esteja interessado em estudar o efeito no resultado final provocado por alguma alteração em algum destes parâmetros.

O “Projeto Árvores Filogenéticas” trabalha internamente com as árvores usando uma estrutura matricial de dados, normalmente empregada para a descrição de grafos, por fornecerem também informações a respeito de hierarquia (ancestrais e descendentes), ao contrário das implementações tradicionais de algoritmos genéticos, que trabalham com estruturas de dados do tipo lista (apenas informações ordenadas linearmente). Maiores informações sobre estruturas de dados podem ser encontradas no Apêndice C. Estes grafos são manipulados pelo software usando sua representação baseada em matriz de adjacências. Isto permitiu o desenvolvimento e emprego dos operadores especiais de mutação utilizados pelo algoritmo genético, baseados em trocas de colunas desta matriz de adjacências. Esta abstração simplificou bastante sua implementação.

O método de busca Máxima Verossimilhança é bastante sensível em relação ao modelo de substituição de bases empregado. Isto ficou bem evidenciado ao longo dos estudos pelos resultados experimentais apresentados no Capítulo 5. Neste trabalho, foram estudados apenas os modelos de substituição de bases de Jukes-Cantor (1969), Felsenstein (1981) e Kimura (1980), por serem os mais freqüentemente encontrados na literatura. O uso de outros modelos

de substituição de bases fica indicado como perspectiva futura de pesquisa e poderá ser explorado em conjunto com outros métodos de busca alternativos em relação ao da Matriz de Distâncias e Máxima Verossimilhança, aqui apresentados.

6.3 Contribuições

Neste trabalho, foi possível ratificar que a Computação Evolutiva constitui uma ferramenta de boa qualidade para bioinformática. Sua capacidade de explorar vastos espaços de solução, aliada ao seu elevado desempenho, produziram excelentes resultados durante as simulações computacionais.

As principais contribuições apresentadas por este trabalho foram:

- Apresentar as técnicas de Computação Evolutiva como uma poderosa ferramenta para ajudar no problema de encontrar uma boa árvore filogenética, a partir de uma seqüência de bases analisadas, devido ao seu elevado poder de exploração e explotação de vastos espaços de solução;
- Fornecer uma visão holística do problema de reconstrução das árvores filogenéticas, desde a analise das seqüências de bases até o ajuste dos comprimentos dos ramos, que foi implementado baseado em um método numérico de primeira ordem conhecido como gradiente;
- Mostrar que, para este problema, o uso de codificação de uma árvore em uma matriz de adjacências é uma boa alternativa, por preservar as informações relativas à hierarquia (ancestrais e descendentes), o que não ocorre com as estruturas de dados tradicionalmente utilizadas, normalmente baseadas em listas;
- Desenvolver novos operadores especiais de mutação para o algoritmo genético, baseados em trocas de colunas da matriz de adjacências utilizada para representar a árvore filogenética candidata;
- Oferecer aos pesquisadores da área uma ferramenta computacional (Projeto Árvores Filogenéticas) com interface gráfica amigável e uma série de recursos já mencionados em seções anteriores.

6.4 Temas sugeridos como perspectivas futuras da pesquisa

Como sugestões para extensões desta pesquisa podemos citar os tópicos a seguir:

- Ajuste automático dos parâmetros usados nos critérios de geração de novas populações (reprodução, mutação e *crossover*);

Alguns trabalhos estudados que exploraram estas características obtiveram ganhos relevantes relativos a desempenho do software e qualidade final do resultado esperado. Parece ser promissor unir estas características aos outros tópicos apresentados nesta dissertação, tais como os novos operadores de mutação e as estruturas de dados internas do Projeto Árvores Filogenéticas.

- Auto-ajuste nos critérios empregados na seleção dos novos elementos a serem levados para a próxima geração em um algoritmo genético;

Assim como os parâmetros relativos às taxas de mutação, parece ser bastante promissor o emprego de parâmetros auto-ajustáveis relativos aos critérios de seleção.

- Trabalhar com várias populações candidatas, simulando nichos ecológicos, com migrações eventuais entre eles;

Este item parece ser particularmente interessante devido ao fato das populações perderem sua capacidade de evoluir após algum tempo de processamento. Trabalhar com populações distintas e permitir migrações eventuais poderia restituir a capacidade de evolução a estas populações.

- Utilizar as técnicas de co-evolução associadas com o conceito de nichos ecológicos;

Este item parece ser promissor devido aos motivos esclarecidos no item anterior e também pelo fato da co-evolução aumentar as pressões seletivas, acelerando o processo de busca mas também acelerando o processo de perda da capacidade de evoluir das diferentes populações.

- Técnicas alternativas de agrupamento de dados para acelerar o processo de convergência do algoritmo genético;

Existem outros métodos que não foram explorados por não estarem dentro dos propósitos inicialmente estabelecidos para esta pesquisa. Seria interessante estender o potencial da ferramenta computacional desenvolvida para poder trabalhar também com outros métodos tais como *neighbor-joining* e máxima parcimônia. Como eles se baseiam em princípios diferentes, a comparação do resultado final seria enriquecedora para os pesquisadores da área. Poderia

inclusive tornar possível a validação dos resultados obtidos com um método pelo emprego dos demais.

- Utilizar algum método de segunda ordem para o ajuste dos comprimentos dos ramos em comparação com o de primeira ordem utilizado;

Existe a possibilidade de se empregar métodos de segunda ordem para o ajuste nos comprimentos dos ramos das árvores. Esperamos que o emprego deste tipo de método de otimização traga ganhos significativos tanto em termos de desempenho da ferramenta quanto em termos de qualidade final dos dados, por usar informações de segunda ordem que geralmente são mais ricas que as de primeira ordem, utilizadas ao longo de nossa pesquisa de mestrado.

- Utilizar modelos alternativos de substituição de bases e comparar com os três utilizados (Jukes-Cantor, 1969; Felsenstein, 1981 ou Kimura, 1980);

Os três modelos de substituição de bases utilizados em nossa pesquisa são os mais utilizados nesta área mas não são os únicos. Existem alguns outros que não foram analisados mas que podem trazer resultados muito interessantes para o problema de reconstrução de árvores filogenéticas. Felsenstein e Kimura possuem outros modelos com dois parâmetros que poderiam ser acrescentados ao escopo da ferramenta computacional para comparação dos resultados. É sabido que trabalhar com mais parâmetros pode aumentar a qualidade do resultado final mas também aumenta o risco de piorar o resultado devido à possibilidade de inserir maior grau de incerteza nos cálculos. Seria muito interessante comparar os resultados obtidos com os modelos de um parâmetro, utilizados em nossa pesquisa de mestrado, com os resultados obtidos com os modelos de dois parâmetros, para poder analisar os custos e os ganhos obtidos.

- Utilizar outro método de busca, diferente dos dois implementados (Matriz de Distâncias e Máxima Verossimilhança);

Devido aos propósitos inicialmente estabelecidos para esta pesquisa de mestrado, não foi possível explorar todas as técnicas disponíveis nem todos os métodos existentes para reconstrução das árvores filogenéticas. Exploramos aqui apenas os métodos conhecidos como Matriz de Distâncias e Máxima Verossimilhança. Existem alguns outros muito bons que também

poderiam ser explorados e que provavelmente trarão resultados interessantes sob o ponto de vista científico.

- Realizar experimentos utilizando computador com processadores paralelos;

Durante todo o desenvolvimento do Projeto Árvores Filogenéticas não foi possível utilizar outro tipo de computador. Sempre usamos um computador comum com apenas um processador e portanto não investigamos os ganhos que poderiam ser obtidos com o uso de máquinas com vários processadores paralelos. Como os algoritmos genéticos processam populações com diversos indivíduos, seria interessante tentar dividir uma fração destes elementos para cada processador e analisar os resultados em função do tempo de processamento. Imaginamos que este tipo de paralelismo possa acelerar muito o processo de busca de uma boa árvore filogenética.

- Promover uma interação continuada com pesquisadores da área de biologia molecular e genética.

Os cientistas de cada área têm muito a contribuir com os de outras. Citamos aqui apenas duas mas poderiam ser muitas outras. Os conhecimentos que os cientistas de algumas áreas podem obter facilmente podem ser obtidos pelos cientistas de outras com um custo muito maior. Durante nossa pesquisa de mestrado, foi fundamental o apoio e a ajuda do Professor Sérgio Furtado dos Reis pelo seu amplo conhecimento na área de Biologia Molecular, sem o qual não seria possível atingir os resultados que conseguimos em tão pouco tempo de pesquisa. Este tipo de experiência deve ser incentivado sempre que possível.

Referências Bibliográficas

Aarts, E. & Korst, J. 1989. *Simulated Annealing and Boltzmann Machines – A Stochastic Approach to Combinatorial Optimization and Neural Computing*, John Wiley & Sons.

Aarts, E. & Verhoeven, M. 1997. Operations Research, in Bäck, T., Fogel, D. B. & Michalewicz, Z. Handbook of Evolutionary Computation, Oxford University Press.

Aarts, E. & Lenstra, J. K. 1997. Local Search in Combinatorial Optimization, Wiley.

Amabis, J. M. & Martho, G. R. 1997. *Biologia – Populações – Genética, Evolução e Ecologia*, Editora Moderna Ltda., vol. 3.

Angelini, P. J. 1993. *Evolutionary Algorithms and Emergent Intelligence*. Doctoral Dissertation, Ohio State University.

Atmar, W. 1994. *Notes on the simulation of evolution*. IEEE Trans. Neural Networks NN-5:130-47.

Bäck, T.; Fogel, D. B. & Michalewicz, Z. 1997. *Handbook of Evolutionary Computation*. Institute of Physics Publishing and Oxford University Press.

Bertoni, A. & Dorigo, M. 1993. *Implicit Parallelism in Genetic Algorithms*, Artificial Intelligence, 61(2): 307-314.

Bezdek, J. C. 1994. *What is computational Intelligence?* Computational Intelligence: Imitation Life.

Burks, C.; Cinkosky, M. J.; Fischer, W. M.; Gilna, P.; Hayden, J. E. D.; Keen, G. M.; Kelly, M.; Kristofferson, D. & Lawrance, J. 1992. *GenBank*, Nucleic Acids Research. 20:2065-2069.

Cavalli-Sforza, L. L. & Edwards, A. W. F. 1964. *Analysis of Human Evolution*.In Proc. 11th Intl. Cong. Genet., pp. 923-933. Pergamon, New York.

Cavalli-Sforza, L. L. & Edwards, A. W. F. 1967. *Phylogenetic analysis: Models and estimation procedures*. Am. J. Hum. Genet. 19:233-257.

Dawkins, R. 1976. The Selfish Gene, Oxford: Oxford University Press.

Day, W. H. E. 1987. *Computational complexity of inferring phylogenies from dissimilarity matrices*. Bull. Math. Biol. 49:461-467.

de Castro, L. N. 1998. *Análise e Síntese de Estratégias de Aprendizado para Redes Neurais Artificiai*s, Dissertação de Mestrado, Faculdade de Engenharia Elétrica e de Computação, Unicamp.

de Castro, L. N., Iyoda, E. M., Von Zuben, F. J. & Gudwin, R. 1998. *Feedforward Neural Network Initialization: an Evolutionary Approach*, Proceedings of the Vth Brazilian Symposium on Neural Networks, vol. 1, pp. 43-48.

Eck, R. V. & Dayhoff, M. O. 1966. *Atlas of protein sequence and structure*. National Biomedical Research Foundation. Silver Spring, MD.

Edwards, A. W. F. & Cavalli-Sforza, L. L. 1963. *The reconstruction of evolution*. Heredity 18:553.

Eiben, A. E., Hinterding, R. & Michalewicz, Z. 1999. *Parameter Control in Evolutionary Algorithms*. IEEE Transactions on Evolutionary Computation, 3(2): pp. 124-141.

Enciclopédia Mirador Internacional. 1995. 9:4592-4599.

Eshelmann, L. J., Caruana, R. A. & Schaffer, J. D. 1989. *Biases in the Crossover Landscape*, in Schaffer, J. (ed.), Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, 10-19.

Everitt, B. S. 1993. *Cluster Analysis*. Halsted Press, third edition.

Felsenstein, J. 1981. *Evolutionary trees from DNA sequences: A maximum likelihood approach*. J. Mol. Evol. 17:368-376.

Felsenstein, J. 1984. *PHYLIP: Phylogeny inference package, version 2.51.* University of Washington, Seattle, WA.

Felsenstein, J. 1990. *PHYLIP Manual Version 3.3* University Herbarium, University of California, Berkeley.

Felsenstein, J. 1997. An alternating least squares approach to inferring phylogenies from pairwise distances. Syst. Biol. 46:101-111.

Fitch, W. M. 1971. *Toward defining the course of evolution: Minimum change for a specific tree topology*. Syst. Zool. 20:406-416.

Fitch, W. M. & Margoliash, E. 1967. *Construction of phylogenetic trees*. Science 155:279-284.

Fogel, L. J., Owens, A. J. & Walsh, M. J. 1966. *Artificial Intelligence Through Simulated Evolution*, John Wiley.

Fogel, D. B. 1994. *An Introduction to Simulated Evolutionary* Computation, IEEE Transactions on Neural Networks, 5(1): pp. 3-14.

Fogel, D. B. 1999. *Evolutionary Computation – Toward a New Philosophy of Machine Intelligence*, 2ª edição, IEEE Press.

Foster, J. 2001. *Evolutionary Computarion*. Nature Reviews / Genetics 2:428-436.

Goldberg, D. E. 1989. *Genetic Algorithm in Search, Optimization and Machine Learning*. Addison-Wesley. Reading. MA.

Goloboff, P. 1994. *NONA: A tree searching program*. Program and documentation, available at ftp.unt.edu.ar/pub/parsimony.

Goloboff, P. 1999. *Analysing large data sets in reasonable times: Solutions for composite optima*. Cladistics 15:415-428.

Goloboff, P.; Farris, J. & Nixon, K. 1999. *T.N.T.: Tree Analysis Using New Technology*. Program available at www.cladistics.com.

Goloboff, P. & Farris, J. 2001. *Metods for Quick Consensus Estimation*. Cladistics 17:S26-S34.

Gordon, A. D. 1981. *Classification: Methods for the Exploratory Analysis of Multivariate Data*. Chapman and Hall.

Gruau F. 1994. *Genetic Micro Programming of Neural Networks*, in Kinnear Jr., K. (ed.), Advances in Genetic Programming, The MIT Press, pp. 495-518.

Guerrero, J. A. S.; Suárez, L. L. & Gudwin, R. 1999. *Análise da Importância de Parâmetros num Algoritmo Genético por meio de sua Aplicação no Aprendizado de uma Rede Neural*. II ENIA:1-6.

Haeckel, E. 1866. *Generelle Morphologie der Organismen*. George Riemer, Berlin.

Hartl, D. L. & Clark, A. G. 1989. *Principles of Population Genetics*, Sinauer.

Higgins, D. G.; Fuchs, R.; Stoehr, P. J. & Camerson, G. N. 1992. *The EMBL Data Library*. Nucleic Acids Research. 20:2071-2074.

Holland, J. H. 1973. *Genetic algorithms and the optimal allocation of trials*. SIAM J. Comput. 2:88-105.

Holland, J. H. 1992. *Adaptation in Natural and Artificial Systems*. MIT Press. Cambridge.

Horowitz, E.& Sahni, S. 1984. *Fundamentos de Estruturas de Dados*. Editora Campus.

Iyoda, E. M. 2000. *Inteligência computacional no projeto automático de redes neurais híbridas e redes neurofuzzy heterogêneas*. Dissertação de Mestrado, Faculdade de Engenharia Elétrica e de Computação, Unicamp.

Jain, A. K. & Dudes, R. C. 1988. *Algorithms for Clustering Data*. Prentice Hall.

Jan, P. 1995. *Coevolutionary Computation*. Artificial Life 2:355-375.

Janis, D. A. & Wheeler, W. 2001. *Efficiency of Parallel Direct Optimization*. Cladistics 17:S71-S82.

Jukes, T. H. & Cantor, C. R. 1969. *Evolution in protein molecules*. Academic Press, New York.

Kaufman, L. & Rousseeuw, P. J. 1990. *Finding Groups in Data: An Introduction to Cluster Analusis*. John Wiley and Sons, Inc.

Kimura, M. 1980. *A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences*. *Journal of Molecular Evolution* **16**:111-120.

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

Koza, J. R.; Bennet III, F. H.; Andre, D.; Keane, M. A. & Dunlap, F. 1997. *Automated Synthesis of Analog Electrical Circuits by Means of Genetic Programming*, IEEE Transactions on Evolutionary Computation, 1(2): pp. 109-128

Kumar, S. 1996. *A stepwise algorithm for finding minimum evolution tress.* Mol. Biol. Evol. 13:584-593.

Lewis, Paul, O. 1998. *A Genetic Algorithm for Maximum-Likelihood Phylogeny Inference Using Nucleotide Sequence Data*. Mol. Biol. Evol. 15:277-283.

Manber, U. 1989, *Introduction to Algorithms. A Creative Approach*. Addison-Wesley Publishing Company.

Massart, D. L.; Vandeginste, B. G. M.; Bydens, L. M. C.; de Jong, S.; Lewis, P. J. & Smeyers-Verbeke, J. 1997. *Genetic algorithms and other global search strategies*. Handbook of Chemometrics and Qualimetrrics: Part A. Elsevier, Amsterdam. 805-845.

Matsuda, H.; Olsen, G. J.; Hagstrom, R.; Overbeek, R. & Kaneda, Y. 1993. *Implementation of a Parallel Processing system for Inference of Phylogenetic Trees*. IEEE. 280-283.

Matsuda, H. 1995. *Construction of phylogenetic trees from amino acid sequences using a genetic algorithm*. Procedings of Genome Informatics Workshop. Universal Academy Press. Tokio, pp. 19-28.

Matsuda, H. 1996. *Protein phylogenetic inference using maximum likelihood with a genetic algorithm*. Pacific Symposium on Biocomputing. World Scientific, London, pp. 512-523.

Matsuda, H.; Yoshikawa, T.; Tabe, T.; Kishinami, R. & Hashimoto, A. 1999. *On the Implementation of a Phylogenetic Tree Database*. IEEE, 42-45.

Merz, P. & Freisleben, B. 1999. A Comparison of Memetic Algoritms, Tabu Search, and Ant Colonies for the Quadratic Assignment Problem, Congress on Evolutionary Computation, vol. 3, pp. 2063-2070.

Michalewicz, Z. 1996. *Genetic Algorithms + Data Structures = Evolution Programs*, 3ª edição, Springer.

Michalewicz, Z. & Schoenauer, M. 1996. *Evolutionary Algorithms for Constrained Parameter Optimization Problems*, Evolutionary Computation, 4(1): 1-32.

Moilanen, A. 2001. *Simulated Evolutionary Optimization and Local Search: Introduction and Application to Tree Search*. Cladistics 17:S12-S25.

Moscato, P. 1999. Memetic Algorithms: A Short Introduction, in Corne, D., Dorigo, M. & Glover, F. (eds.) New Ideas in Optimization, McGraw-Hill, pp. 219-234.

Moscato, P. & Norman, M. G. 1992. A memetic approach for the travelling salesman problem – implementation of a computational ecology for optimisation on message-passing systems, Proceedings of the International Conference on Parallel Computing and Transputer Applications, Amsterdam, IOS Press.

Muse, S. V. & Weir, B. S. 1992. *Testing for equality of evolutionary rates*. Genetics 132:269-276.

Nei, M. 1987. *Molecular evolutionary genetics*. Columbia University Press, New York.

Nei, M.; Tajima, F. & Tateno, Y. 1983. *Accuracy of estimated phylogenetic trees from molecular data.* II. Gene frequency data. J. Mol. Evol. 19:153-170.

Nei, M. & Kumar, S. 2000. *Moledular Evolution and Phylogenetics*. Oxford University press.

Nixon, K. 1999. *The parsimony ratchet, a new method for rapid parsimony analysis*. Cladistics 15:407-414.

Olsen, G. J.; Matsuda, H.; Hagstrom, R. & Overbeek, R. 1993. *fastDNAml: A Tool for Construction of Phylogenetic Trees of DNA Sequences Using Maximum Likelihood*. Compute Applications in Biological Sciences.

Ockham, W. 1964. "*Philosophicall Writings*", Tradução Philotheus Boehner. Indianápolis, Bobbs-Merrill.

Potter, M. & DeJong, K. 2000. *Cooperative coevolution: An architecture for evolving coadapted subcomponents*. Evolutionary Computation, 8(1):1-29.

Radcliffe, N. J. & Surry, D. P. 1994. “Formal Memetic Algorithms”, Evolutionary Computing: AISB Workshop, Ed: T. Forgaty, Springer-Verlag.

Rasmussem, E. 1992. *Information Retrieval*, chapter Clustering Algorithms, pages 419-442. Prentice Hall.

Rechenberg, I. 1973. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Holzboog.

Reijmers, T.H.; Wehrens, R.; Daeyaert, F. D.; Lewi, P. J. & Buydens, L. M. C. 1999. *Using genetic algorithms for the construction of phylogenetic trees: application to G-protein coupled receptor sequences*. Biosystems. 49:31-43.

Rice, K.; Donoghue, M. & Olmstead, R. 1997. Analysing large data sets: rbcL 500 revisited. Syst. Biol. 46:554-563.

Rzhetsky, A & Nei, M. 1993. *Theoretical foundation of the minimum-evolution method of phylogenetic inference*. Mol. Biol. Evol. 10:1073-1095.

Saitou, N.& Nei, M. 1987. *The neighbor-joining method: A new method for reconstructing phylogenetic trees*. Mol. Biol. Evol. 4:406-425.

Schwefel, H. –P. 1995. *Evolution and Optimum Seeking*, John Wiley.

Skourikhine, A. 2000. *Phylogenetic Tree Reconstruction Using Self-Adaptive Genetic Algorithm*. Proceedings of the 1st IEEE International Symposium on Bioinformatics and Biomedical Engineering, Arlington, Virginia, USA. 129-134.

Syed, O. 1995. *Applying Genetic Algorithms to Recurrent Neural Networks for Learning Network Parameters and Architecture*. Master Thesis, Case Western Reserve University.

Silva JR., C. & Sasson, 1996. S. Biologia, Editora Saraiva, vol. 3.

Sneath, P. H. A & Sokal, R. R. 1973. *Numerical taxonomy*. Freeman, San Francisco, CA.

Sokal, R. R. & Sneath, P. H. A. 1963. *Principles of Numerical Taxonomy*. Freeman, San Francisco, CA.

Swofford, D. L. 1993. *PAUP: Phylogenetic Analysis Using Parsimony, version* 3.1. Program and documentation, Laboratory of Molecular Systematics, Smithsonian Institute, Washington, DC.

Swofford, D. L. & Olsen, G. J. 1990. *Phylogeny reconstruction. Molecular Systematics*. Sinauer, Sunderland, MA.

Swofford, D. L.; Olsen, G. J.; Wadell, P. J. & Hillis, D. M. 1996. *Phylogeny inference. Molecular systematics*. Sinauer, Sunderland, MA, pp. 407-514.

Syswerda, G. 1989. Uniform *Crossover in Genetic Algorithms*, em Schaffer, J. D. (ed.), Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, 2-9.

Takezaki, N. & Nei, M. 1996. *Genetic distances and reconstruction of phylogenetic trees from microsatellite DNA*. Genetics 144:389-399.

Tateno, Y.; Nei, M. & Tajiima, F. 1982. *Accuracy of estimated phylogenetic trees from molecular data*. I. Distantly related species. J. Mol. Evol. 18:387-404.

Weir, B. S. 1996. *Genetic Data Analysis* II. Sinauer, Sunderland, MA.

Wu, C-I. & Li, W-H. 1985. *Evidence for higher rates of nucleotides substitution in rodentes than in man*. Proc. Natl. Acad. Sci. U.S.A. 82:1741-1745.

Yang, Z. 1999. PAML: Phylogenetic analysis by maximum likelihood, ver. 2.0 University College London, London.

Zadeh, L. A. 1965. “Fuzzy Sets”, *Information and Control*, 8: 338-353, re-impresso em Klir, G. J. & Juan, B. (eds.), *Fuzzy Sets, Fuzzy Logic and Fuzzy Systems: Selected Papers by Lotfi A. Zadeh*, World Sci, 1996.

Índice de Autores

Aarts & Korst (1989) 15
Aarts & Lenstra (1997) 43
Aarts & Verhoeven (1997) 43
Amabis & Martho (1997) 8
Angelini (1993) 19
Atmar (1994) 3, 7, 9, 10, 11
Bäck *et al.* (1997) 3, 7, 9, 18, 19, 26, 77, 109, 149
Bezdek (1994) 2
Burks *et al.* (1992) 66
Cavalli-Sforza & Edwards (1967) 53, 108, 110, 169
Dawkins (1976) 40
Day (1987) 81
de Castro (1998) 96, 97
de Castro *et al.* (1998) 24
Eck & Dayhoff (1966) 57
Edwards & Cavalli-Sforza (1963) 54, 57
Enciclopédia Mirador Internacional (1995) 47, 48, 107
Eshelman *et al.* (1989) 21
Everitt (1993) 50
Felsenstein (1981) 5, 60, 62, 63, 74, 78, 101, 102, 104, 105, 106, 107, 108, 109, 112, 114, 134, 135, 143
Felsenstein (1984) 63, 64
Felsenstein (1990) 67, 73, 99
Felsenstein (1997) 66
Fitch & Margoliash (1967) 54, 69
Fitch (1971) 57
Fogel (1994) 9, 13, 17, 26
Fogel (1999) 9, 10, 11
Fogel *et al.* (1966) 13

Foster (2001)....33, 34
Goldberg (1989)....25, 77, 149
Goloboff & Farris (2001)....70
Goloboff (1994)....71
Goloboff (1999)....71
Goloboff *et al*. (1999)....71
Gordon (1981)....50
Gruau (1994)....13
Guerrero *et. al*. (1999)....24
Haeckel (1866)....1
Hartl & Clarck (1989)....9
Higgins *et al*. (1992)....67
Holland (1973)....12, 16, 77
Holland (1992)....16, 23, 77
Horowitz & Sahni (1984)....176, 177
Iyoda (2000)....3
Jain & Dudes (1988)....50
Jan (1995)....25
Janis & Wheeler (2001)....71
Jukes-Cantor (1969)....5, 62, 63, 64, 74, 78, 101, 102, 104, 106, 107, 108, 109, 112, 114, 134, 135, 143
Kaufman & Rousseeuw (1990)....50
Kimura (1980)....5, 63, 64, 74, 78, 101, 102, 105, 106, 107, 108, 109, 112, 114, 134, 135, 143
Koza (1992)....13
Koza *et al.* (1997)....13
Kumar (1996)....56
Lewis (1998)....1, 2, 68, 81, 109
Manber (1989)....34, 35, 173, 174, 175, 176, 178, 179, 180
Massart *et al*. (1997)....77
Matsuda (1995)....73, 77, 109
Matsuda (1996)....68, 77
Matsuda *et al*. (1993)....66, 71, 72
Matsuda *et al.* (1999)....1
Merz & Freisleben (1999)....42, 43

Michalewicz & Schoenauer (1996) 22, 23, 24
Michalewicz (1996) 13, 15, 16, 17, 22, 23, 24, 26, 149
Moilanen (2001) 72
Moscato & Norman (1992) 40, 41
Moscato (1999) 41, 44
Muse & Weir (1992) 102
Nei & Kumar (2000) 48, 50, 51, 52, 53, 54, 55, 56, 57, 59, 60, 61, 79, 80, 81, 83, 107, 110, 169
Nei (1987) 48
Nei *et al.* (1983) 51
Nixon (1999) 71, 72
Ockham (1964) 57
Olsen *et al*. (1993) 67
Potter & DeJong (2000) 25
Radcliffe & Surry (1994) 40, 43
Rasmussen (1992) 50
Rechenberg (1973) 13
Reijmers *et al.* (1999) 2, 48, 69, 85
Rice *et al*. (1997) 71
Rzhetsky & Nei (1993) 54, 55, 56
Saitou & Nei (1987) 55, 56, 69
Schwefel (1995) 13
Silva & Sasson (1996) 40
Skourikhine (2000) 25, 70
Sneath & Sokal (1973) 51
Swofford & Olsen (1990) 1
Swofford (1993) 71
Swofford *et al*. (1996) 49, 50, 56, 57, 59, 60, 63, 64
Syed (1995) 27, 28, 30, 31, 32, 33
Syswerda (1989) 21
Takezaki & Nei (1996) 51
Tateno *et al.* (1982) 48
Weir (1996) 1, 2, 48, 49, 50, 55, 56, 57, 58, 59, 61, 64, 77, 78, 98, 99, 104, 133, 135, 143, 151
Wu & Li (1985) 102

Yang (1999) 73
Zadeh (1965) 2

Apêndice A

Manual do Usuário do Projeto Árvores Filogenéticas

Projeto Árvores Filogenéticas

Manual do Usuário

Versão 1.0.3

APÊNDICE A MANUAL DO USUÁRIO DO PROJETO ÁRVORES FILOGENÉTICAS129

A.1. INTRODUÇÃO....131
A.2. HISTÓRICO....131
A.3. DESCRIÇÃO GERAL....132
A.7. CONTROLE DE ACESSO EXTERNO....132
A.8. REQUISITOS DE DESEMPENHO....133
A.9. REQUISITOS FUNCIONAIS....133
A.10. EQUIPAMENTO CLIENTE (MÍNIMO)....142
A.11. TELAS DO SISTEMA....143
A.12. UMA OPERAÇÃO COMPLETA, PASSO A PASSO....157
A.13. CONFIGURAÇÃO DO ARQUIVO COM AS SEQÜÊNCIAS DE DNA....164

A.1. Introdução

A.1.1. Objetivos

Este documento tem o objetivo de apresentar instruções sobre a forma de utilização do Projeto “Árvores Filogenéticas“ para avaliação de seqüências de DNA a fim de encontrar as árvores filogenéticas que melhor exprimem os graus de parentesco entre as espécies.

Este trabalho foi baseado em Weir (1996). Grande parte das informações necessárias para implementar o método da Máxima Verossimilhança no **Projeto Árvores Filogenéticas**, foram encontradas em seu livro.

A.1.2. Público alvo deste documento

Este documento se destina às pessoas envolvidas em pesquisa genética que requer a realização de inferência filogenética.

A.2. Histórico

Data	Versão	Responsável	Alteração
16/02/2001	1.0.0	Oclair Prado	Versão original deste documento
26/02/2001	1.0.1	Oclair Prado	Inclusão da operação passo a passo
10/08/2001	1.0.2	Oclair Prado	Inclusão da opção de gravar evolução de fitness ao longo das gerações
11/11/2001	1.0.3	Oclair Prado	Alterações realizadas após revisão do Prof. Dr. Von Zuben

A.3. Descrição Geral

Este sistema nasceu como uma forma de contribuição para o trabalho de pesquisadores que buscam as árvores filogenéticas que melhor expliquem os graus de parentesco entre as espécies, baseados na análise de seqüência de DNA.

A.4. Produtos do Sistema

Este sistema fornece as melhores árvores filogenéticas para um grupo de seqüências de DNAs.

A.5. Funcionalidades do Sistema

- Busca as árvores filogenéticas que melhor expliquem os graus de parentesco entre os elementos pesquisados;
- Efetua comparações entre os três modelos mais conhecidos: Jukes-Cantor (1969), Felsenstein (1981) e Kimura (1980);
- Lê e grava arquivos em formato texto com as seqüências de DNA dos elementos em análise;
- Permite interromper a busca, gravar os dados e continuar em outra data;
- Permite alteração de grande parte de seus parâmetros de busca, podendo ser customizado para melhor adequação aos objetivos do pesquisador.

A.6. Características do usuário

Todos os usuários do sistema devem ter noções básicas do uso de microcomputador com sistema operacional semelhante ao Windows da Microsoft.

A.7. Controle de Acesso Externo

O sistema não possui módulo específico para tratamento de segurança de acesso de usuários.

A.8. Requisitos de Desempenho

A quantidade de elementos que o sistema pode tratar é limitada apenas pela máquina utilizada. Um equipamento com processador rápido e boa quantidade de memória RAM é recomendado.

A.9. Requisitos Funcionais

A.9.1. Busca as árvores filogenéticas que melhor expliquem os graus de parentesco entre os elementos pesquisados

A.9.1.1. Descrição

Conforme descrito por Weir (1996), a busca das melhores árvores filogenéticas depende muito do modelo de substituição de bases a ser utilizado. Atualmente, os principais modelos são os de Jukes-Cantor (1969), Felsenstein (1981) e Kimura (1980).

Este sistema permite que a busca seja realizada pelo método da Matriz de Distâncias (método não baseado em modelo) ou pelo método da Máxima Verossimilhança (método baseado em modelo).

A.9.1.2. Processamento

Primeiro, deve-se preparar o arquivo com os elementos a serem pesquisados com seus nomes e seqüências de DNA. Depois de se iniciar o sistema, deve-se clicar no botão “Iniciar processamento”. Esta busca será realizada baseada no modelo solicitado na tela de parâmetros e será realizada com apoio de técnicas de Computação Evolutiva, o que permite maior agilidade e rapidez na busca de boas soluções dentre um grande número de candidatas. No entanto, não há garantia de obtenção da solução ótima.

Os resultados são apresentados nas abas “Matriz de distâncias” e “Árvores” da tela principal do sistema.

A.9.1.3. Critério de parada adotado pelo algoritmo genético

Nesta versão do **Projeto Árvores Filogenéticas**, o algoritmo genético encerra suas iterações quando as três árvores com os maiores índices de *fitness* forem iguais. Isto equivale a dizer que ele para quando os três melhores elementos forem idênticos. As simulações

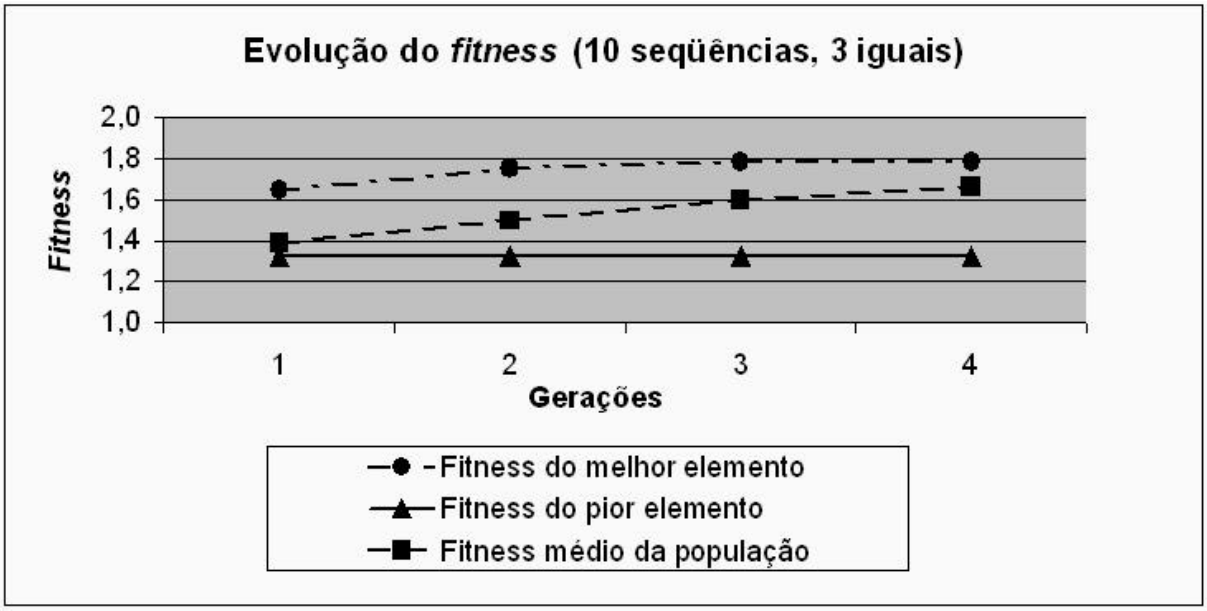
apresentadas a seguir, mostram que a população perde sua capacidade de evoluir após algum tempo de processamento.

Em todos os casos analisados, este ponto de máximo sempre foi encontrado quando os três melhores elementos se tornaram iguais. A partir deste momento, nenhum elemento da população conseguiu ultrapassar os melhores elementos. A única alteração visível após este ponto, foi a melhora dos outros elementos da população que tenderam a se aproximar do ponto de máximo encontrado.

Todas as simulações foram realizadas mantendo a população fixa em 100 elementos para a população total e 100 elementos para a população intermediária. A única diferença que se percebe entre os dois conjuntos de simulações, foi o número da iteração em que os três melhores elementos foram encontrados. Este número variou em função do número de folhas (seqüências) analisadas e não em função do critério de parada selecionado.

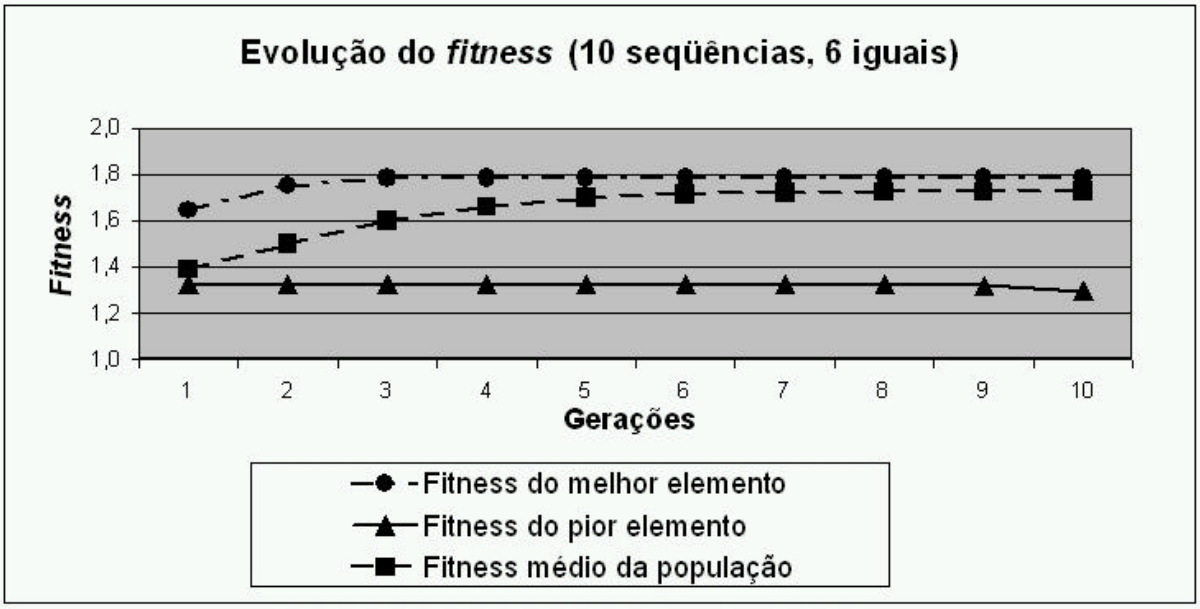
Resultados encontrados com dez folhas (seqüências)

Na figura a seguir, pode ser observado que o ponto onde a população perde sua capacidade de evoluir coincide com o ponto onde os três melhores elementos são iguais, ou seja, na terceira geração. Este comportamento se repetirá ao longo das próximas figuras.

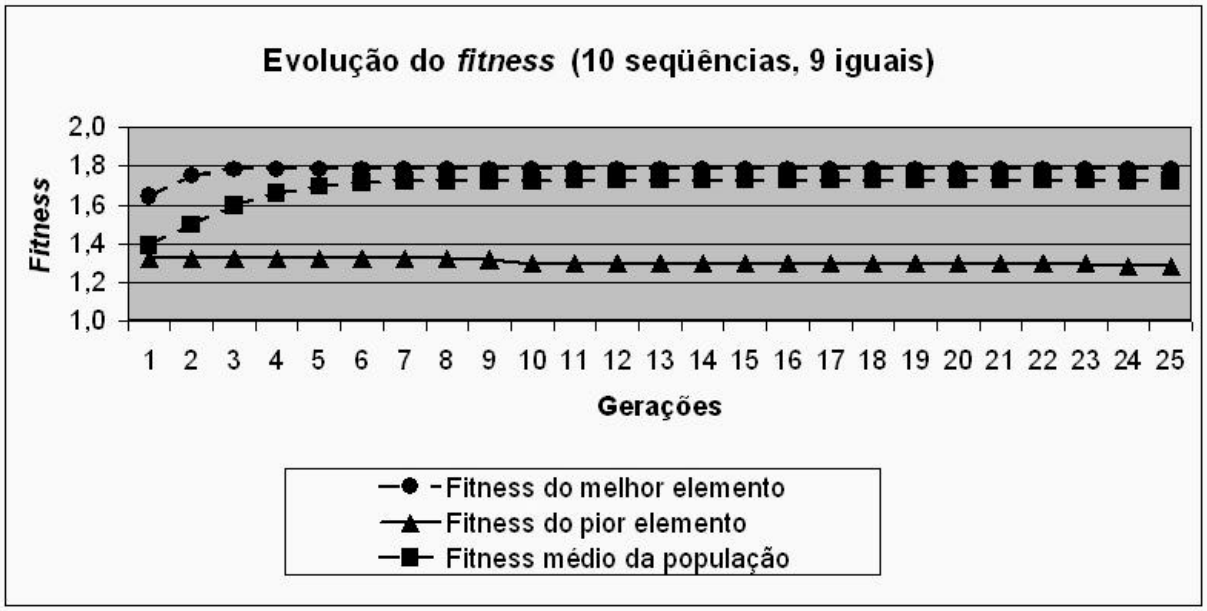


A próxima figura ressalta que não é possível melhorar o resultado após o ponto em que os três melhores elementos são iguais. Nesta simulação, o critério de parada foi alterado para

que o algoritmo genético encerrasse suas iterações quando os seis melhores elementos fossem iguais. Novamente, após a terceira geração, a população parou de evoluir.

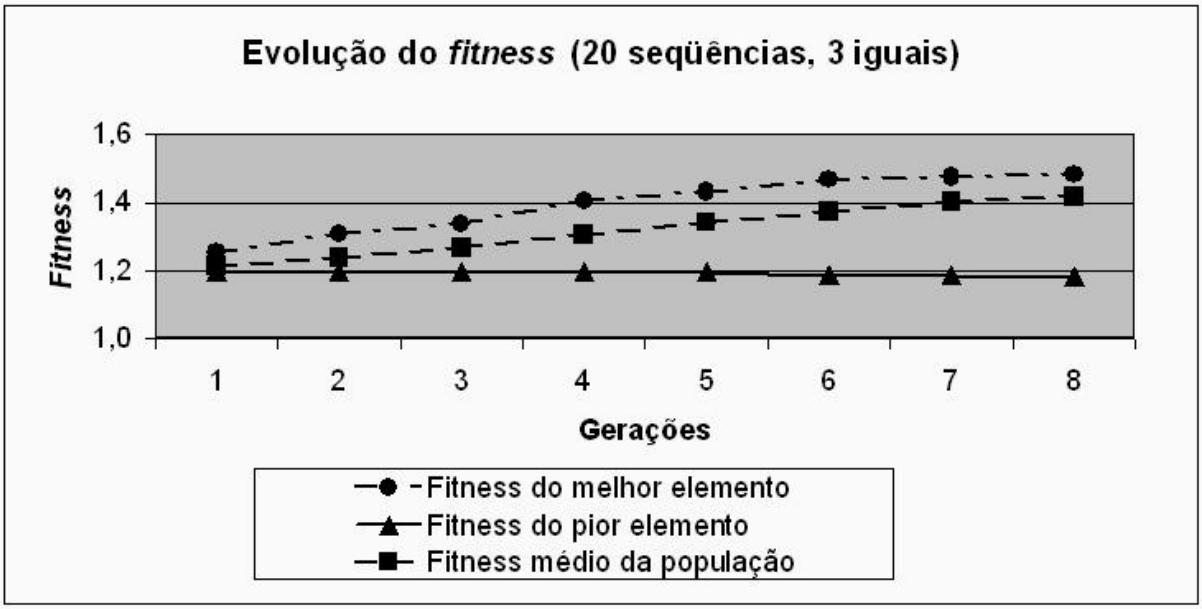


Na figura a seguir, observamos que, após a terceira geração, a população não mais evoluiu. Tivemos apenas um aumento na quantidade de elementos que ocuparam a primeira posição (*fitness* mais elevado). Nesta simulação, o critério de parada foi alterado para que o algoritmo genético encerrasse suas iterações quando os nove melhores elementos fossem iguais.

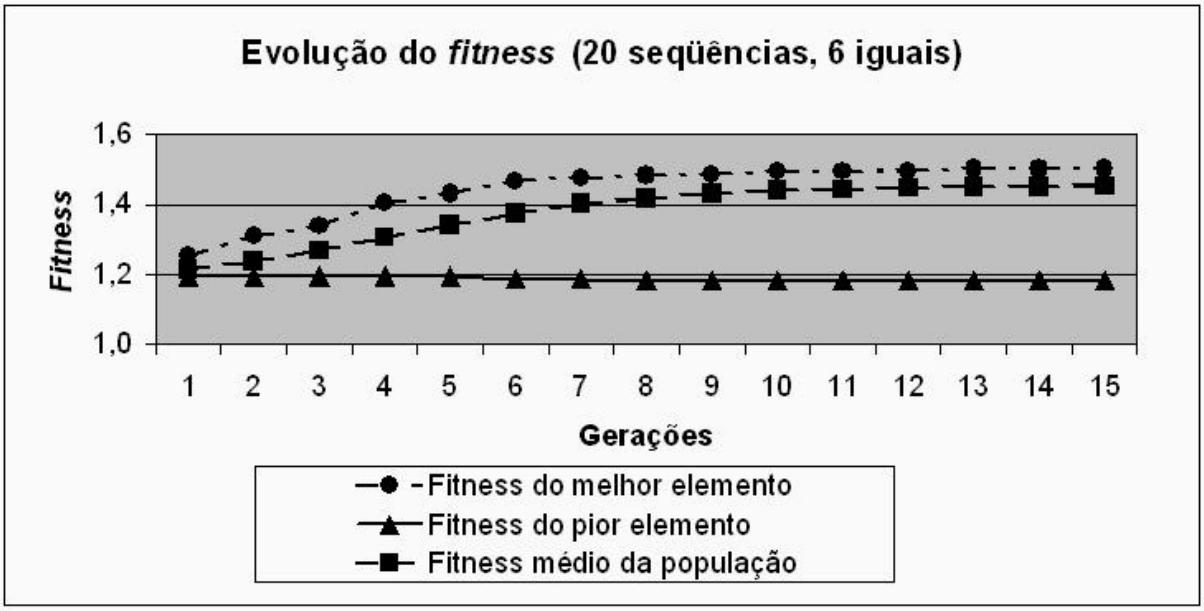


Resultados encontrados com vinte folhas (seqüências)

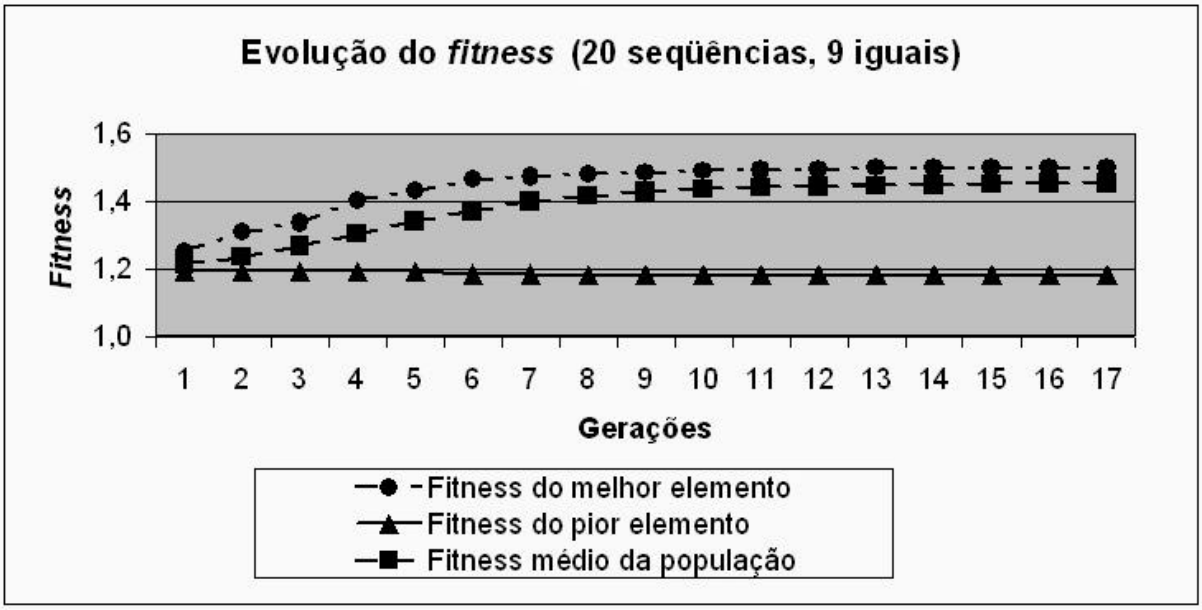
Na figura a seguir, pode ser observado que o ponto onde a população perde sua capacidade de evoluir coincide com o ponto onde os três melhores elementos são iguais. Este comportamento já foi ressaltado para o caso de dez folhas e se repetirá ao longo das próximas figuras. Neste caso, a população parou de evoluir após a sétima geração.



Assim como foi encontrado no caso das dez folhas, descrito anteriormente, nesta figura visualizamos que não é possível melhorar o resultado após o ponto em que os três melhores elementos são iguais. Nesta simulação, o critério de parada foi alterado para que o algoritmo genético encerrasse suas iterações quando os seis melhores elementos fossem iguais. A única diferença em relação ao resultado apresentado para dez folhas foi a geração em que os três melhores elementos se tornaram iguais. Neste caso, isto ocorreu na sétima geração.



Na figura a seguir, observamos que, após a sétima geração, a população não mais evoluiu. Tivemos apenas um aumento na quantidade de elementos que ocuparam a primeira posição (*fitness* mais elevado). Nesta simulação, o critério de parada foi alterado para que o algoritmo genético encerrasse suas iterações quando os nove melhores elementos fossem iguais.



A.9.1.4. Conclusões sobre o critério de parada adotado pelo algoritmo genético

A análise dos dois conjuntos de simulações permite concluir que o critério de parada adotado apresenta bons resultados, considerando a qualidade do resultado obtido e o tempo de processamento necessário.

Em todas as simulações, tivemos apenas alterações no número de gerações necessárias para atingir o critério de parada. O ponto de máximo se manteve constante, independente de se parar quando encontrar os três, seis ou nove primeiros elementos iguais.

Os gráficos a seguir resumem os resultados encontrados em função da quantidade de gerações necessárias para atingir o critério de parada e do tempo de processamento associado ele.

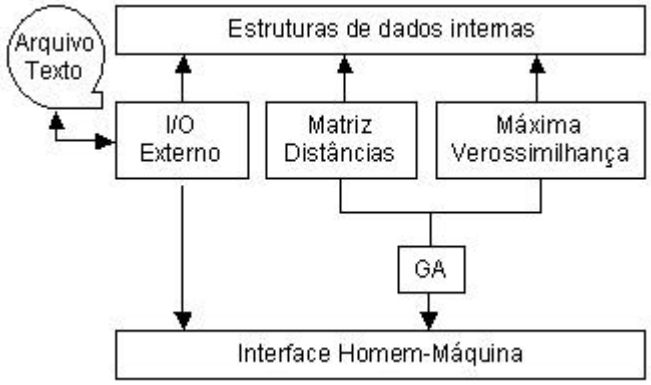
É necessário ressaltar a alteração na escala de tempo do gráfico relativo aos resultados obtidos com as simulações relativas às vinte folhas. O custo em função do tempo de processamento se mostrou muito alto nos dois casos, das dez de das vinte folhas. Isto era um resultado esperado, pois foram necessárias mais iterações (gerações) para que os elementos medianos das populações evoluíssem até alcançar o melhor elemento.

Devidos aos resultados apresentados, o algoritmo genético encerra suas iterações ao quando os três melhores elementos da população forem iguais.

A.9.1.5. Diagrama de blocos do sistema

Esta versão do sistema foi totalmente realizada usando a linguagem Visual Basic 6, da Microsoft.

A figura a seguir apresenta seus blocos funcionais.



Interface Homem-Máquina

A Interface Homem-Máquina é bastante intuitiva, seguindo o padrão Windows de controle de botões e janelas. Esta é a parte da ferramenta que interage diretamente com o usuário. Serve como uma camada destinada a facilitar sua utilização. Através dela, o usuário pode selecionar os arquivos de seqüências a serem analisadas, o método de busca, o modelo de substituição de bases e os diversos parâmetros do algoritmo genético. Também é este módulo que se encarrega de apresentar o resultado final do processamento ao usuário e permite sua impressão e sua gravação em arquivo texto.

I/O Externo

Este é o módulo responsável por administrar as atividades de leitura e gravação em arquivo texto. O sistema deve ler as seqüências de bases a serem analisadas a partir de um arquivo texto e, ao final do processamento, é possível gravar os resultados também em arquivo texto. Este é o módulo que se encarrega de realizar estas operações de entrada e saída de informações externas.

GA

O módulo GA é a parte central do Projeto Árvores Filogenéticas. Nele foi implementado o algoritmo genético que controla todo o processo de busca das árvores filogenéticas. Para o

cálculo do *fitness* de cada elemento, ele utiliza um dos seguintes módulos, dependendo dos parâmetros configurados pelo usuário. Caso seja selecionado o método da matriz de distâncias, ele usa o módulo "Matriz Distâncias", caso seja selecionado o método da máxima verossimilhança, ele utiliza o módulo "Máxima Verossimilhança". Os operadores genéticos, a seleção natural, o critério de parada também fazem parte deste módulo.

Matriz Distâncias

Este módulo calcula o *fitness* das árvores usando o método da matriz de distâncias. Para realizar esta tarefa, ele usa intensamente a teoria dos grafos. Os grafos foram implementados internamente usando sua representação baseada em matriz de adjacências. Isto permitiu obter bom desempenho do sistema, pois os operadores de alterações genéticas foram preparados para realizar suas operações trabalhando sobre as colunas destas matrizes.

Máxima Verossimilhança

Este módulo calcula o *fitness* das árvores usando o método da máxima verossimilhança. Este é o módulo que mais consome tempo de processamento em todo o sistema. Ele é encarregado de calcular a verossimilhança das árvores candidatas. Para esta operação, é necessário otimizar os comprimentos dos ramos da árvore candidata. Isto é realizado com o uso de um método de otimização de primeira ordem, conhecido como Gradiente.

Este módulo será oportunamente convertido para a linguagem C, para melhorar o desempenho global do sistema.

Estruturas de dados internas

Este módulo representa as variáveis e demais estruturas que o sistema utiliza durante o processamento das seqüências de DNA. A mais relevante é a matriz de adjacências que representa a árvore montada a partir das seqüências de DNA analisadas.

Considerações sobre uso de memória em tempo de execução

Como o tamanho da população não varia ao longo da busca, o espaço de memória necessário para o processamento também não sofre alterações ao longo do processamento. Não foram observados excessivos acessos a disco em nossas execuções utilizando um Pentium II com 128 Megabytes de memória RAM.

Considerações sobre a plataforma do sistema operacional

A versão atual do Projeto Árvores Filogenéticas está restrita aos sistemas operacionais do tipo Windows da Microsoft.

A.9.2. Efetua comparações entre os três modelos de substituição de bases mais conhecidos: Jukes-Cantor (1969), Felsenstein (1981) e Kimura (1980)

A.9.2.1. Descrição

A pesquisa pode ser realizada por um dos três modelos, ou pelos três, com posterior avaliação dos resultados. A comparação é realizada pelo cálculo do "*Likelihood Ratio Test* (*LRT*)" (Weir, 1996) dos melhores resultados encontrados em cada modelo.

A.9.2.2. Processamento

O sistema pesquisa as árvores que melhor expliquem os graus de parentesco entre os elementos usando o modelo de Felsenstein (1981), e em seguida calcula os comprimentos dos ramos da melhor árvore pelos modelos de Jukes-Cantor (1969) e de Kimura (1980).

Os resultados são apresentados na tela de "Comprimentos" do sistema.

A.9.3. Lê e grava arquivos em formato texto com as seqüências de DNA dos elementos em análise

A.9.3.1. Descrição

Para maior versatilidade, os dados são gravados em formato texto seguindo a sintaxe descrita no final deste manual.

A.9.3.2. Processamento

Para realizar o processamento, o sistema necessita que os dados sejam informados na ordem especificada no final deste manual. Como as informações são gravadas em formato texto, é possível disponibilizá-las usando qualquer editor de texto, o que simplifica o processo de geração de novas massas de dados para testes.

A.9.4. Permite interromper a busca, gravar os dados e continuar em outra data

A.9.4.1. Descrição

Como este processamento pode ser demorado, principalmente quando a quantidade de elementos a ser pesquisada for muito grande, é possível que o processo de busca seja interrompido e seus dados possam ser gravados em disco para posterior reinício da busca.

A.9.4.2. Processamento

Durante o processo de busca, é permitido ao usuário interromper a execução do programa. Ele pode gravar os dados atuais em arquivo e recuperar este arquivo em outro dia para dar seguimento aos trabalhos de sua pesquisa. Para estas operações, foram criados os botões de “Leitura” e “Gravação” de arquivos, além dos botões de “Iniciar” e “Parar” a busca.

A.9.5. Permite alteração de grande parte de seus parâmetros de busca, podendo ser customizado para melhor adequação aos objetivos do pesquisador

A.9.5.1. Descrição

Como este sistema foi construído baseado em Computação Evolutiva, ele requer a definição de parâmetros de busca e seleção de candidatos. Ele foi preparado com alguns valores *default,* mas todos eles podem ser alterados para melhor atender às necessidades do pesquisador. Em alguns casos, os valores *default* podem não ser os mais adequados e precisam ser alterados para acelerar o processo de busca.

A.9.5.2. Processamento

Antes de iniciar o processamento, ou entre um processamento e outro, é possível acionar a tela de parâmetros e alterar seus valores. Deve-se tomar muito cuidado com estas alterações pois o efeito delas pode comprometer os resultados. Como a busca é realizada usando técnicas de Computação Evolutiva, um ajuste, ainda que de pequeno valor, pode ter grande impacto no resultado do processamento. Uma maneira de se minimizar os riscos é alterar somente um parâmetro de cada vez e estudar os resultados.

A.10. Equipamento Cliente (mínimo)

1. Características: microprocessador Intel Pentium II, 300 Mhz, 128 Mbytes RAM, 1Gb de disco;

2. Sistema Operacional: Windows.

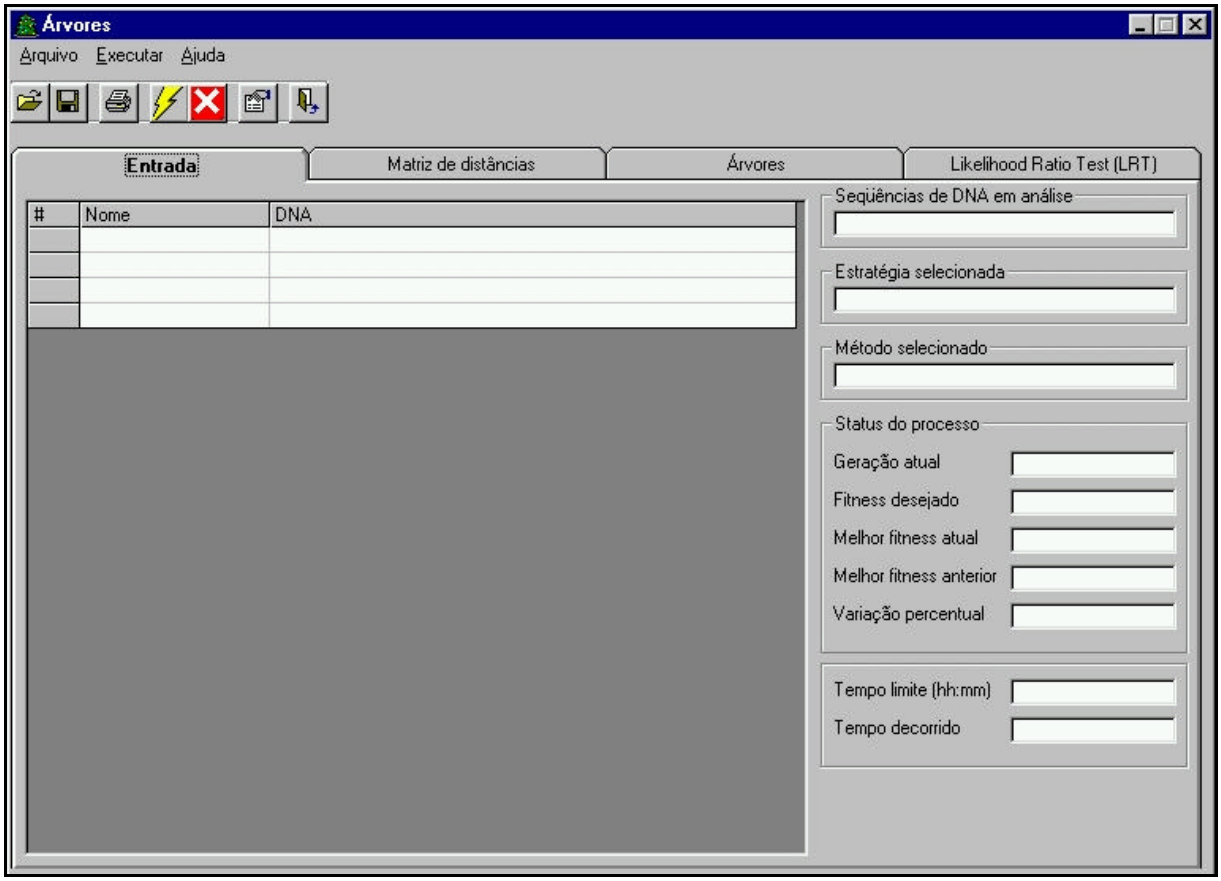
Esta configuração se mostrou suficiente para processar árvores com até 100 folhas (ou espécies). Deve ser enfatizado que o número de folhas a ser analisado é limitado apenas pela quantidade de memória RAM disponível. O espaço em disco recomendado é para armazenar as seqüências e os resultados obtidos.

A.11. Telas do Sistema

A.11.1. Tela Principal

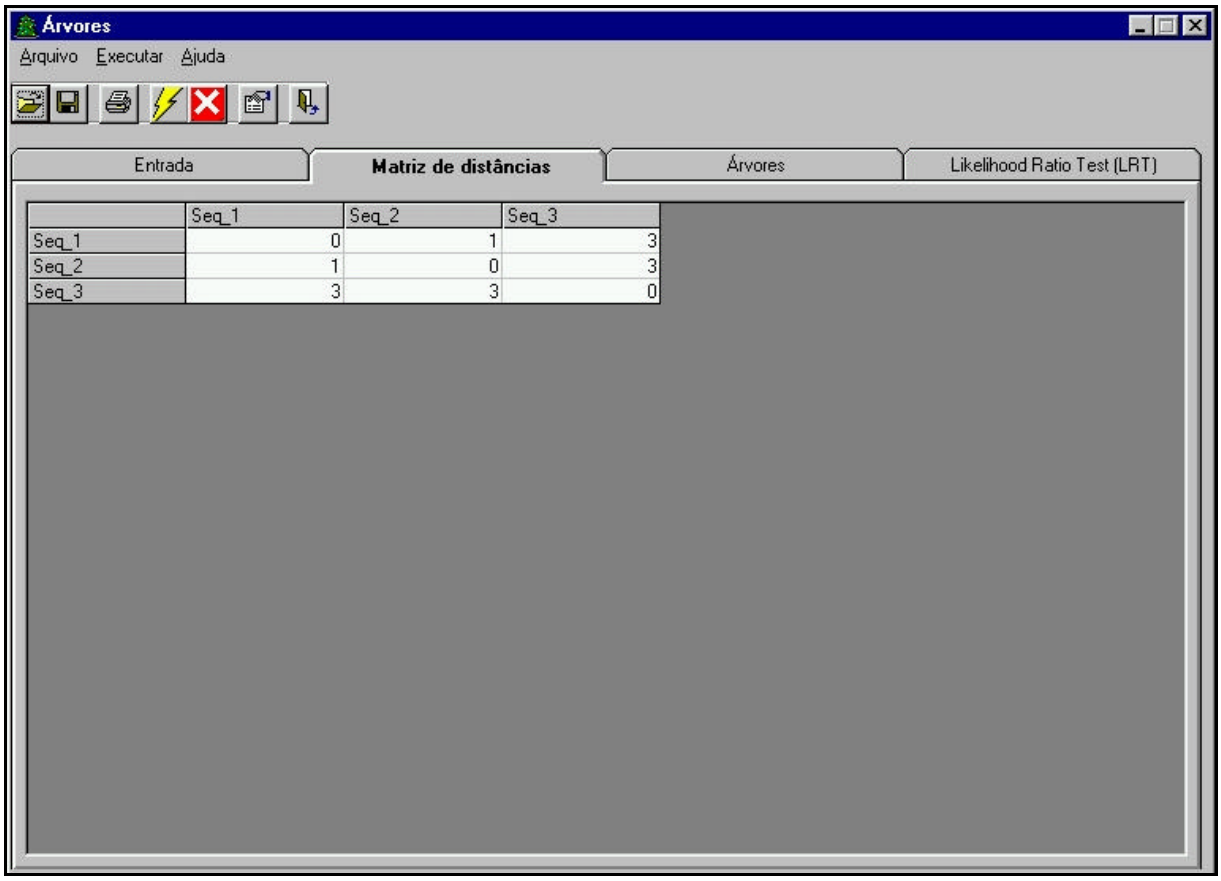
A.11.1.1. Entrada

Esta é a tela principal do sistema. Nesta aba, são apresentados os elementos em análise e algumas informações sobre a busca, tais como estratégia selecionada, tempo decorrido durante o processamento, tempo limite para busca e outras informações relevantes para o acompanhamento da pesquisa.



A.11.1.2. Matriz de Distâncias

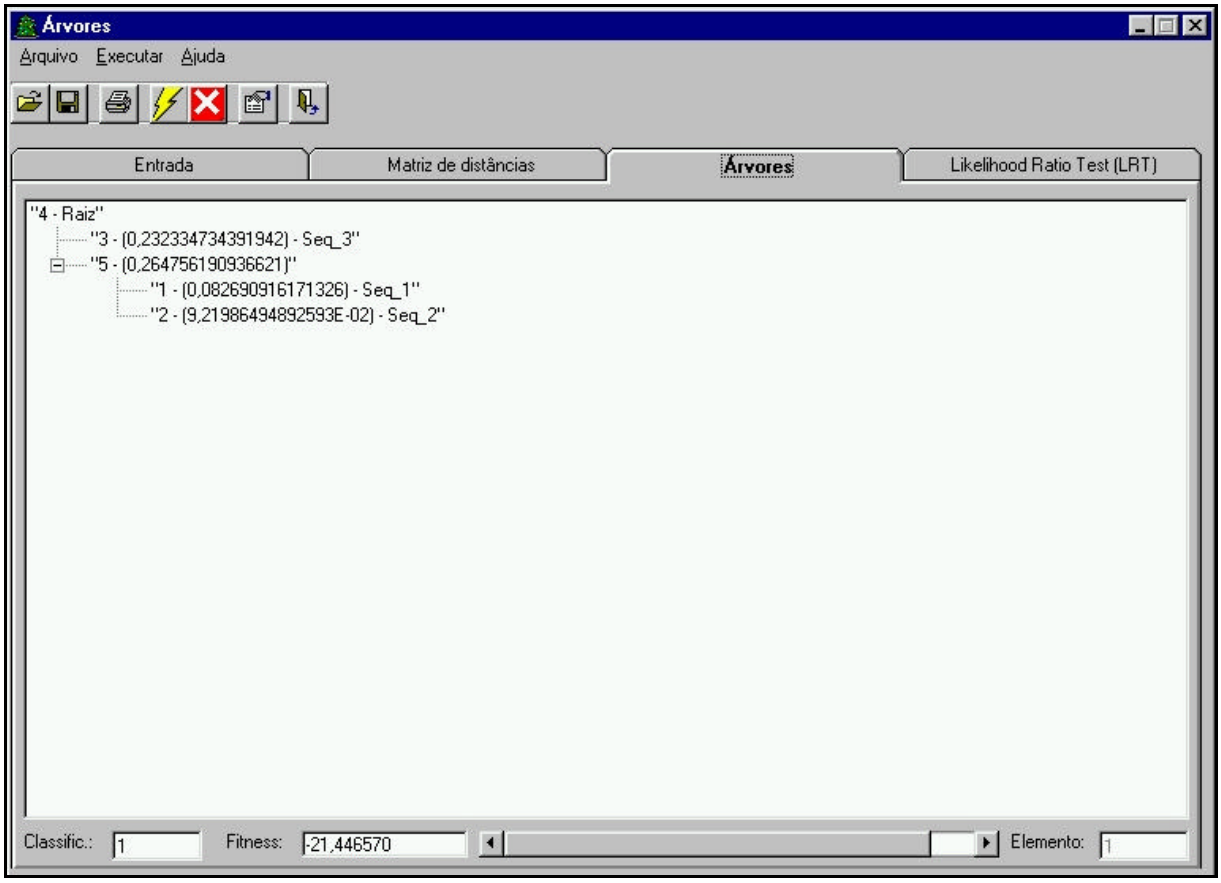
Esta aba apresenta a matriz de distâncias encontradas a partir das seqüências de DNA dos elementos em análise. A comparação é sempre realizada tomando os elemento dois a dois. Com isto, a diagonal principal terá sempre valor zero.



	Seq_1	Seq_2	Seq_3
Seq_1	0	1	3
Seq_2	1	0	3
Seq_3	3	3	0

A.11.1.3. Árvores

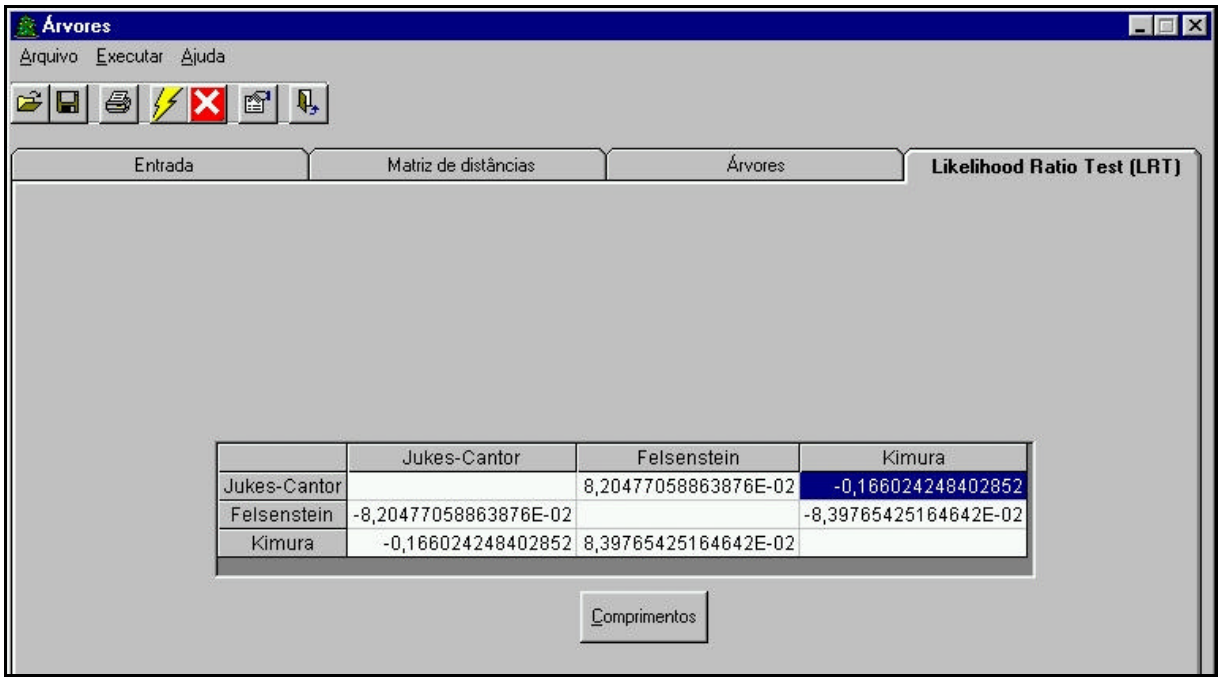
Os resultados da busca serão apresentados nesta aba. Para analisar os melhores elementos, pode-se alterar a árvore apresentada nesta janela digitando a classificação do elemento que se deseja examinar no quadro “Classific.:” na sua parte inferior ou alterando a barra de rolagem de elementos. O *fitness* da árvore apresentada é exibido no quadro “*Fitness*” ao lado do quadro de classificação. As árvores, após sua análise, recebem um valor correspondente ao seu *fitness*. Em seguida, elas são ordenadas e podem ser examinadas usando os controles no final desta tela.



A.11.1.4. *Likelihood Ratio Test* (*LRT*)

Quando o usuário selecionar a estratégia de busca “Comparativa” na tela de parâmetros, os resultados poderão ser visualizados em parte nesta aba e em parte na tela de “Comprimentos”. Quando outra estratégia de busca for selecionada, estas telas ficarão em branco.

A tela de “Comprimentos” pode ser ativada a partir desta aba clicando no botão “Comprimentos”.



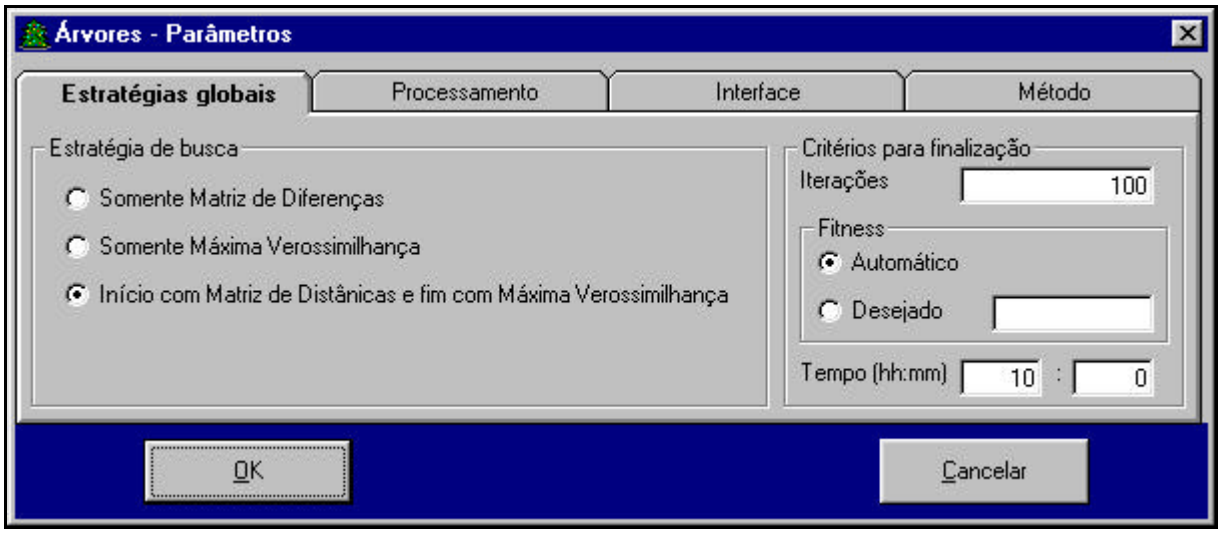
A.11.2. Tela de Parâmetros

A.11.2.1. Estratégias globais

Nesta aba, pode ser selecionada a estratégia de busca a ser adotada, a quantidade limite de gerações a serem criadas pelo sistema durante a busca, o tempo limite para esta busca e o critério de finalização da busca.

O critério de *fitness* automático foi otimizado para atender à maioria das necessidades e é fortemente recomendado. Caso se queira alterar para parar o processamento quando o sistema atingir um *fitness* limite, deve-se tomar o cuidado de observar a alteração implícita contida na estratégia de busca. Quando se utiliza a estratégia da matriz de distâncias, o *fitness* tende a diminuir e é sempre positivo. Quando se utiliza a estratégia da máxima verossimilhança, o

fitness tende a crescer e é sempre negativo. Quando se utiliza a estratégia mista, deve-se utilizar o critério automático.



A.11.2.2. Processamento

Nesta aba, encontram-se os parâmetros específicos para a Computação Evolutiva. Maiores esclarecimentos podem ser encontrados na literatura especializada, como, por exemplo, em Goldberg (1989), Michalewicz (1996) e Bäck *et al.* (1997). Em particular, o Capítulo 2 desta dissertação apresenta os conceitos necessários para identificar o papel de cada parâmetro.

O operador *crossover* para alterações genéticas requer dois elementos para combinar seu DNA e gerar um terceiro com partes das seqüências de DNA dos anteriores. Este tipo de operador não foi implementado neste *software* por necessitar de cuidados especiais a fim de não comprometer a integridade dos novos elementos.

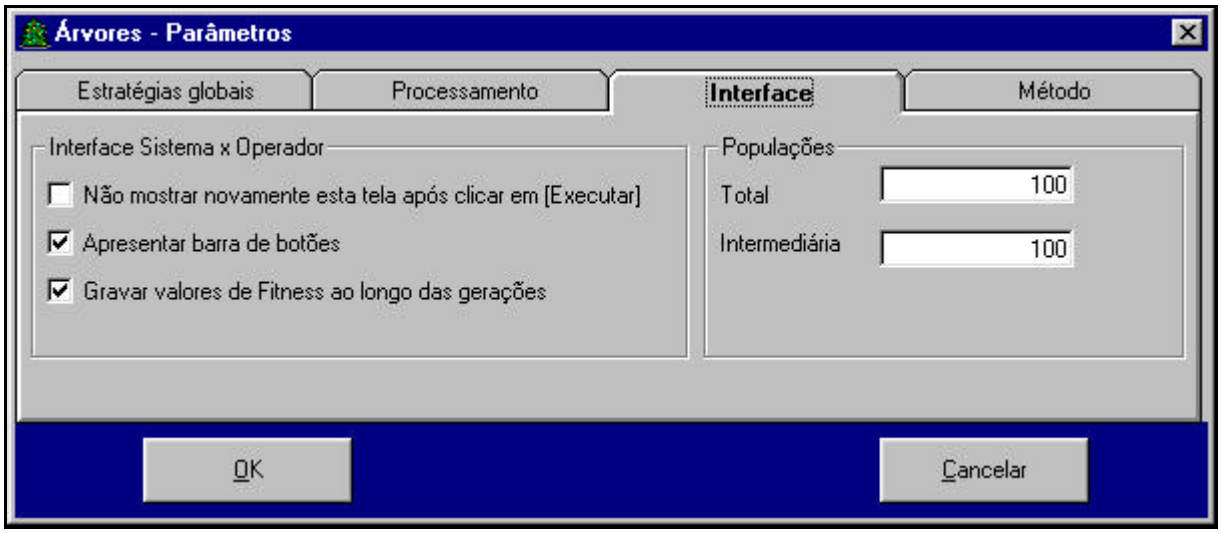
Os operadores implementados nesta versão, foram os relativos à mutação de folhas (ou elementos) e os relativos à mutação entre sub-árvores. Os operadores de mutação executam pequenas transformações na seqüência de DNA de um elemento a fim de gerar um novo descendente. Estes operadores foram implementados baseados em trocas de colunas da matriz de adjacências dos elementos.

O conceito da "roleta" foi aplicado desconsiderando o valor relativo dos elementos dentro da população a ser utilizada. Isto foi realizado utilizando um gerador de números randômicos que são utilizados para selecionar qualquer um dos elementos dentro da população total. Os operadores que atuam sobre o melhor elemento não utilizam este gerador de números randômicos uma vez que eles sempre operam sobre o melhor elemento da população total.

A.11.2.3. Interface

Nesta aba, estão os parâmetros para o limite de elementos nas populações total e intermediária que o sistema deverá gerar durante o processo de busca das melhores árvores. Não existe restrição quanto ao valor de uma ser maior ou menor que o valor da outra. Uma vez que estes valores estejam definidos, eles se mantêm até o final do processamento. Não existe flutuação relativa à quantidade de elementos das populações. A partir da população fixa, os operadores de mutação geram os elementos da população intermediária. Ao final de uma iteração, os elementos das duas populações são avaliados em função de seu *fitness* e são transferidos para a população total da geração seguinte, até que seja encontrada a situação de parada.

Também estão aqui as opções para mostrar ou não a barra de botões e para mostrar ou não a tela de parâmetros no início e reinício do processamento. Outra opção desta tela permite determinar se o sistema deverá armazenar os valores de *fitness*, ao longo das gerações, do melhor elemento, do pior elemento e da média da população.



A.11.2.4. Método

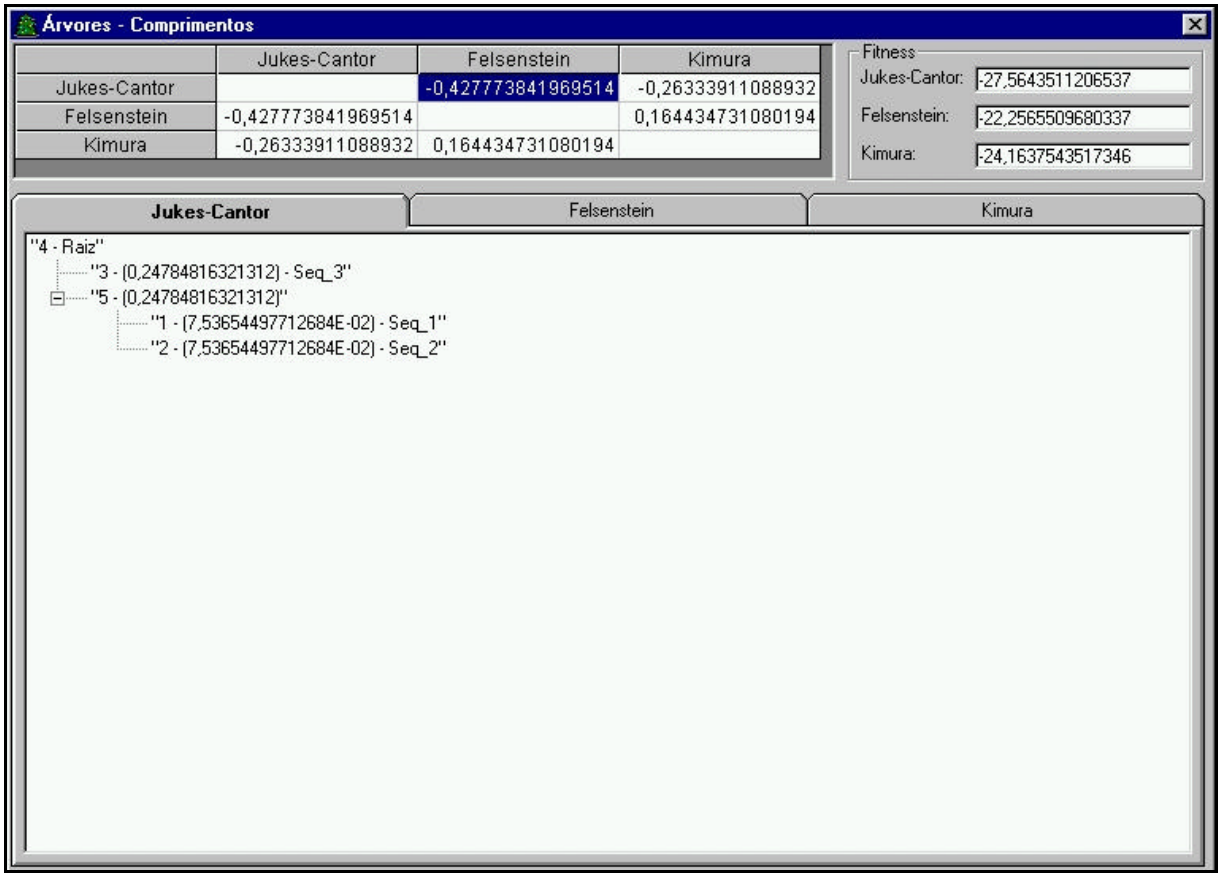
O sistema permite que o usuário selecione um dos três modelos mais conhecidos para a pesquisa, ou selecione o método Comparativo, o qual encontra os comprimentos pelos três modelos e executa a comparação descrita em Weir (1996) como "*Likelihood Ratio Test* (*LRT*)".

Os parâmetros α e β somente serão habilitados quando for selecionado o método "Somente Kimura" ou Comparativo, pois não fazem sentido para os demais.

A.11.3. Tela de Comprimentos

Quando o usuário selecionar a estratégia de busca “Comparativa” na tela de parâmetros, os resultados poderão ser visualizados em parte na aba “*Likelihood Ratio Test* (*LRT*)” da tela principal e na tela de “Comprimentos”, apresentada a seguir. Quando outra estratégia de busca for selecionada, estas telas ficarão em branco.

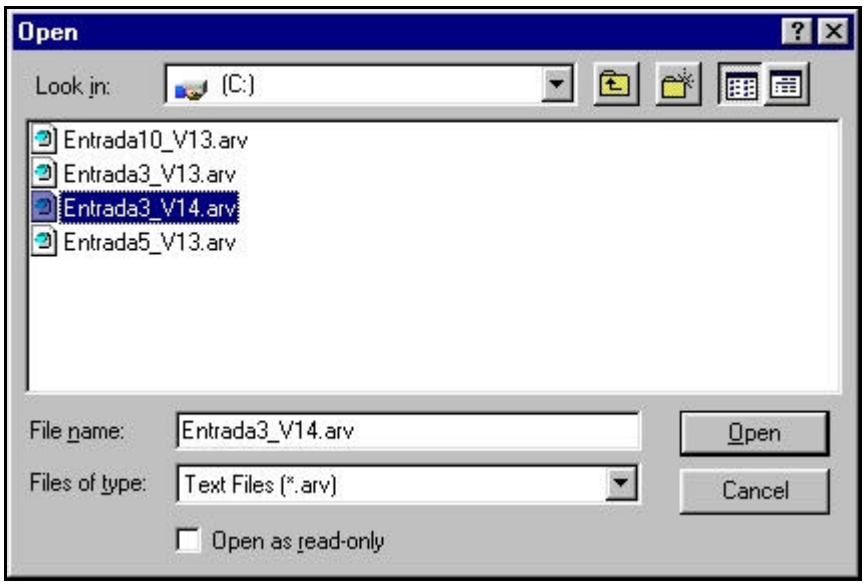
A tela de “Comprimentos” será ativada automaticamente ao final do processamento, ou a partir da aba “*Likelihood Ratio Test* (*LRT*)” da tela principal, clicando no botão “Comprimentos”.

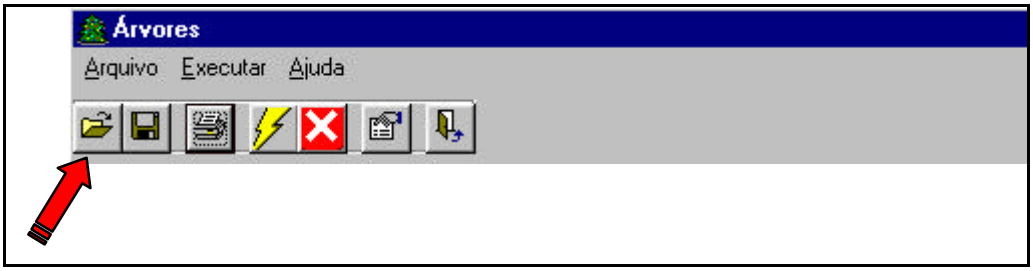


A.11.4. Ler Arquivos

O sistema permite ler arquivos com os dados necessários para a busca.

Todos os elementos devem estar relacionados neste arquivo, junto com suas respectivas seqüências de DNA. Caso existam opções de execução previamente selecionadas e gravadas neste arquivo, elas serão armazenadas na tela de parâmetros e serão utilizadas na continuação da busca.

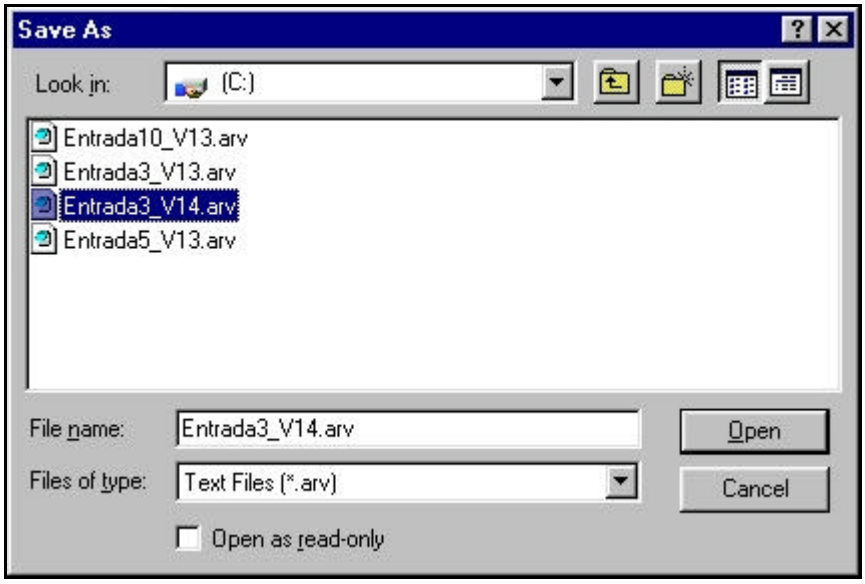




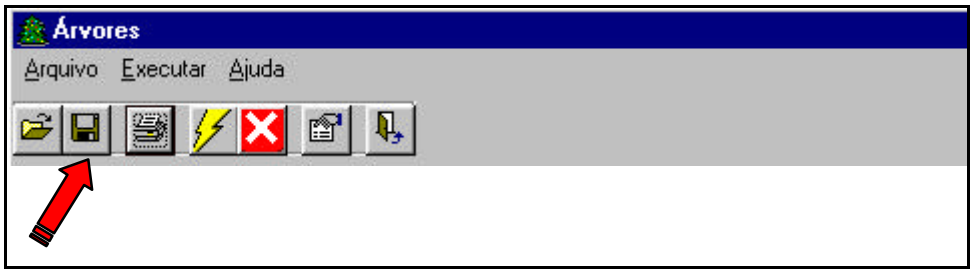
Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Ler arquivo” no menu “Arquivo”.

A.11.5. Gravar Arquivos

O sistema permite que sejam gravados em arquivo todos os dados utilizados até o momento na busca. Todos os elementos e todas as opções definidas na tela de parâmetros serão armazenados para futura consulta ou reinício da execução.



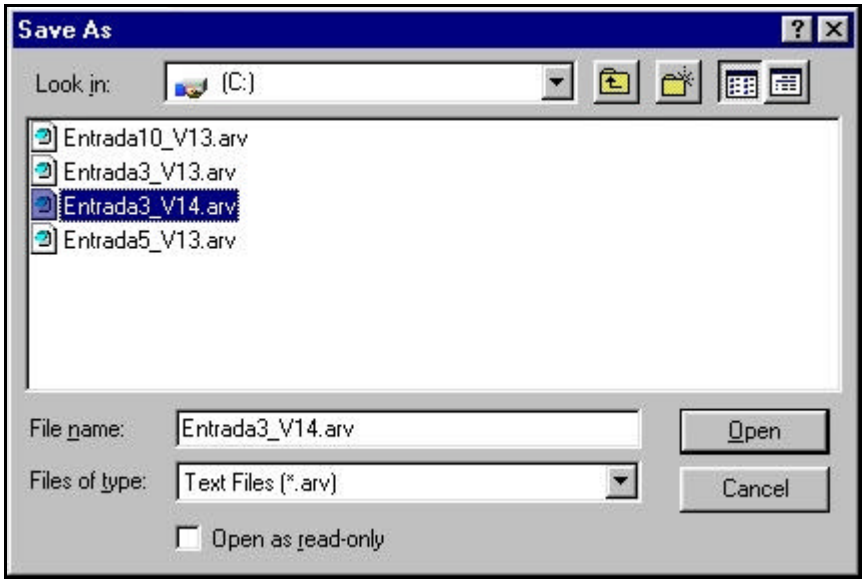
Esta é uma tela padrão do Windows e não requer maiores explicações. Ela permite navegar pelo sistema de arquivos da máquina do usuário para a seleção do diretório onde o arquivo será gravado.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Gravar arquivo” no menu “Arquivo”.

A.11.6. Gravar Evolução

O sistema permite que sejam gravados em arquivo todos os valores de fitness do melhor elemento, do pior elemento e da média da população ao longo de todas as gerações processadas. Este arquivo, do tipo texto, pode ser facilmente importado pelo Excel para criar gráficos sobre o histórico do processo evolutivo.



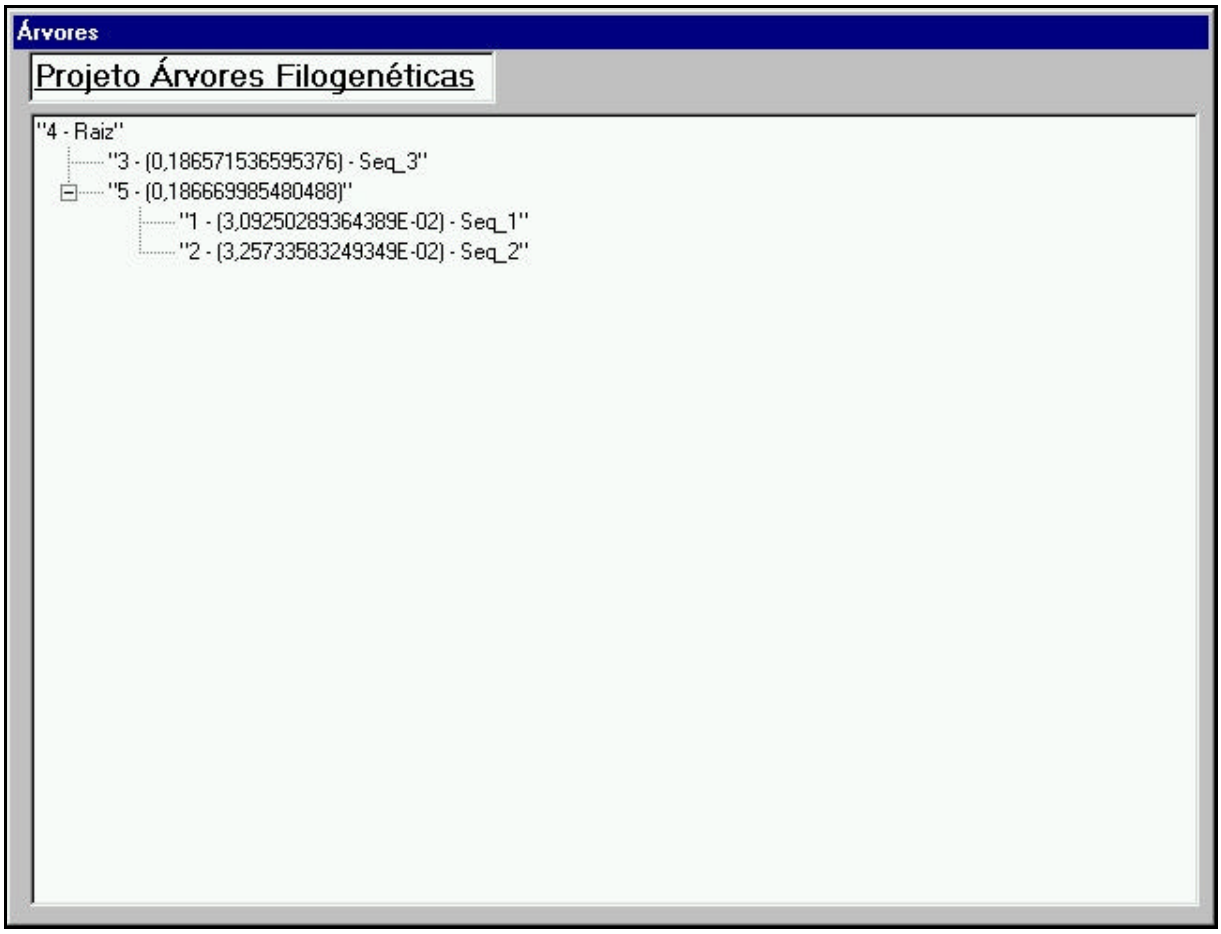
Esta é uma tela padrão do Windows e não requer maiores explicações. Ela permite navegar pelo sistema de arquivos da máquina do usuário para a seleção do diretório onde o arquivo será gravado.

Esta funcionalidade pode ser ativada pela opção “Gravar evolução” no menu “Arquivo”.

A.11.7. Imprimir Árvores

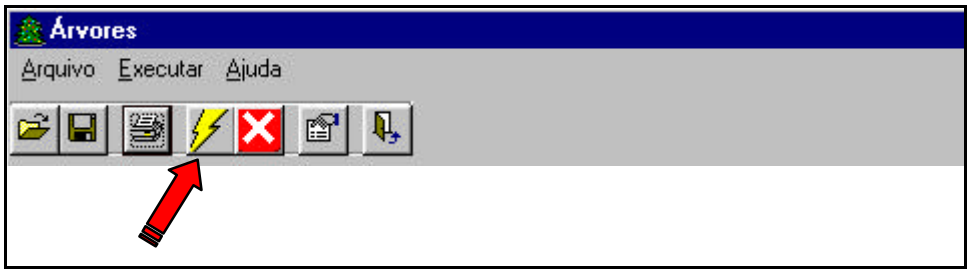
Esta opção é ativada a partir do botão indicado na figura acima.

A tela a seguir não deve ser exibida para o usuário, ela será copiada para a impressora ativa e deve conter a árvore que estiver sendo exibida na aba “Árvores” da tela principal.



A.11.8. Iniciar Busca das Melhores Árvores

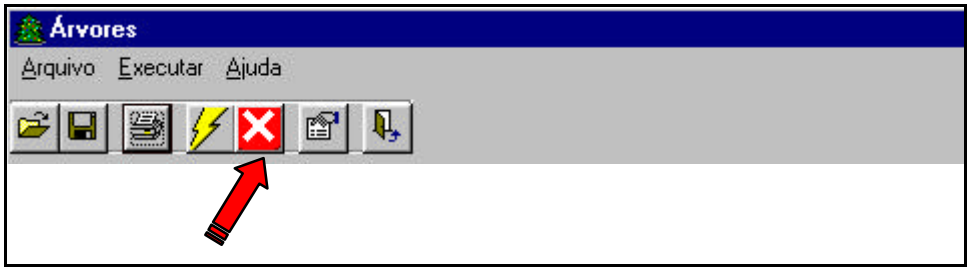
Esta opção inicia o processo de busca pelas melhores árvores para os DNAs dos elementos informados. Esta busca deverá parar quanto encontrar o critério de parada definido na tela de parâmetros, ou quando for acionado o comando para interromper o processo, conforme descrito na próxima seção.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Iniciar” no menu “Executar”.

A.11.9. Parar Busca das Melhores Árvores

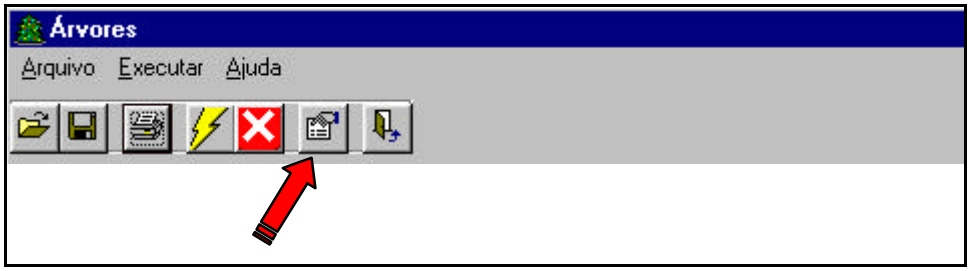
O processo de busca das melhores árvores pode ser longo e alguns ajustes nos parâmetros podem ajudar a melhorar o desempenho do sistema. Para interromper o processamento, foi criado o botão assinalado na figura a seguir. Após o encerramento o processo pode ser reiniciado novamente sem prejuízo das informações encontradas até o momento.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Parar busca” no menu “Executar”.

A.11.10. Modificar Parâmetros

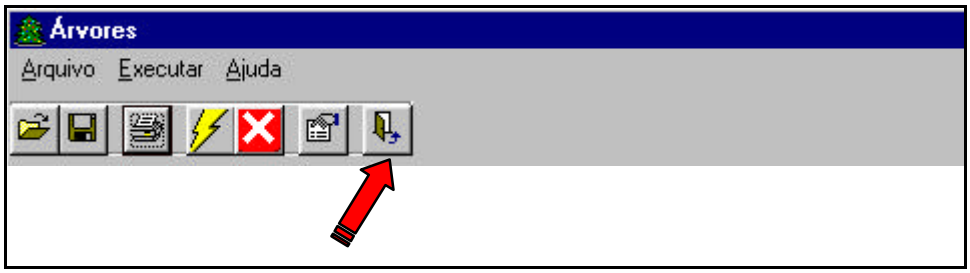
Como este sistema foi construído baseado em Computação Evolutiva, ele é sensível aos parâmetros de busca e seleção de candidatos. Ele foi preparado com alguns valores *default,* mas todos eles podem ser alterados para melhor atender às necessidades do pesquisador. Em alguns casos, os valores *default* podem não ser os mais adequados e precisam ser alterados. A figura a seguir assinala o botão que aciona esta funcionalidade.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Parâmetros” no menu “Executar”.

A.11.11. Encerrar Sistema

Para encerrar o sistema, foi criado o botão assinalado na figura a seguir. O sistema ainda fará mais uma pergunta pedindo confirmação do usuário antes de tirá-lo do ar.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Sair” no menu “Arquivo”.

A.12. Uma Operação Completa, Passo a Passo

O início de operação de um sistema normalmente é um desafio para o qual contamos com pouco apoio e escassa documentação. Esta seção foi preparada para ajudar nos primeiros passos com todos os detalhes e explicações necessárias para cada passo a ser seguido até a obtenção da árvore filogenética e demais resultados associados.

A.12.1. Preparar o Arquivo com as Seqüências de DNA

As seqüências de DNA devem ser escritas em formato texto, obedecendo à sintaxe apresentada no último tópico deste manual.

Os tópicos obrigatórios são somente os dois primeiros do arquivo: **`[Cabec]`** e **`[DNA]`**.

Quando não for possível determinar algum dos parâmetros do sistema deve-se informar apenas até o tópico **`[Parametros]`** (exclusive). A partir deste ponto o sistema assume os valores *default* e permite que o usuário os altere quando necessário.

Como este é o primeiro arquivo a ser processado, ele deve ser criado da seguinte maneira.

Primeiro copie o texto do final deste manual até o tópico **`[Parametros]`** (exclusive) e cole em um editor de texto qualquer. O arquivo resultante deve ficar com o conteúdo a seguir:

```
[Cabec]
Projeto Árvores Filogenéticas
Versao: 1.4 - Demonstração
```

```
Data: 15/jan/2001
Autor: Oclair Prado
Seqüência em análise: DNA de Mitocôndrias
Elementos: 3
[DNA]
Sequencia_1
GTAAATGACA
Sequencia _2
ATAAATGACA
Sequencia _3
CTAAACAACA
```

O campo "`Seqüência em análise:`" deve ser alterado de modo a informar o conteúdo da seqüência de DNA em análise. Esta informação será mostrada na tela principal do aplicativo, já comentada em seções anteriores.

O campo `Elementos:` deve conter a quantidade de elementos a serem estudados. Neste caso temos três elementos.

O tópico `DNA` contém as informações sobre os elementos em análise, sendo a primeira linha reservada para a identificação do elemento e a segunda linha reservada para seu DNA.

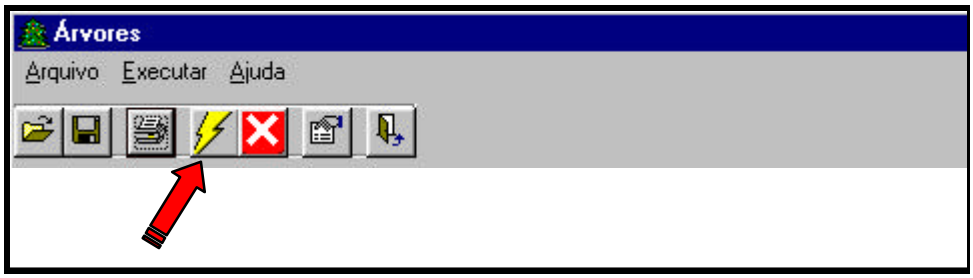
É muito importante para o sistema que este tópico esteja em estreita sintonia com o anterior. Ele deve conter exatamente a quantidade de elementos informada para poder executar a busca.

Após os ajustes necessários é preciso gravar o arquivo. Como o sistema busca primeiro no diretório raiz do *drive* C:, é recomendável que este arquivo seja gravado em "C:\Entrada3_V14.arv". Não existem restrições quanto ao nome do arquivo, além daquelas impostas pelo sistema de arquivos do sistema operacional que estiver sendo utilizado. Ele pode ser escolhido da maneira que melhor apoiar o trabalho do pesquisador.

O número de elementos e a quantidade de sítios de DNA informados não têm limite, mas afetam diretamente o desempenho do sistema, conforme já comentado em seções anteriores.

A.12.2. Iniciar a Busca

Após preparar e gravar o arquivo com as seqüências de DNA, já estamos prontos para iniciar a busca pelas melhores árvores.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Iniciar” no menu “Executar”.

A.12.3. Ajustar os Parâmetros

Em seguida, é apresentada a tela de parâmetros para que o pesquisador possa ajustar os detalhes da busca. Ela aparece preenchida com os valores *default* para acelerar o trabalho.

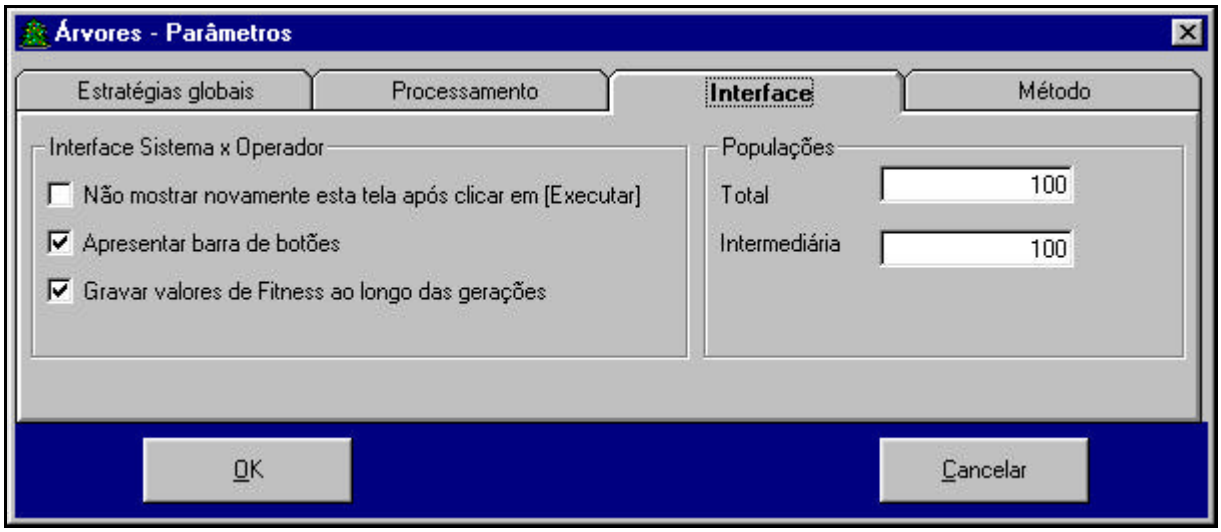
É importante ressaltar que o sistema dará maior prioridade para os parâmetros gravados no arquivo de dados e irá ignorar qualquer configuração diferente da que estiver gravada.

Para alterar os parâmetros de um arquivo existem duas opções:

(a) Carregar o arquivo desejado no sistema, alterar os parâmetros desejados e gravá-lo novamente ou

(b) Carregar o arquivo desejado no sistema, executar o processamento completa ou parcialmente e alterar os parâmetros desejados. Na próxima execução quando o sistema perguntar se deve ou não ler o arquivo de dados basta dizer não e ele irá respeitar os parâmetros alterados. Se for desejado, pode-se gravar o arquivo com os novos parâmetros para futura execução.

Esta tela irá aparecer em todas as execuções, até que seja desmarcada a opção que pede para que ela apareça, conforme descrito na tela a seguir:

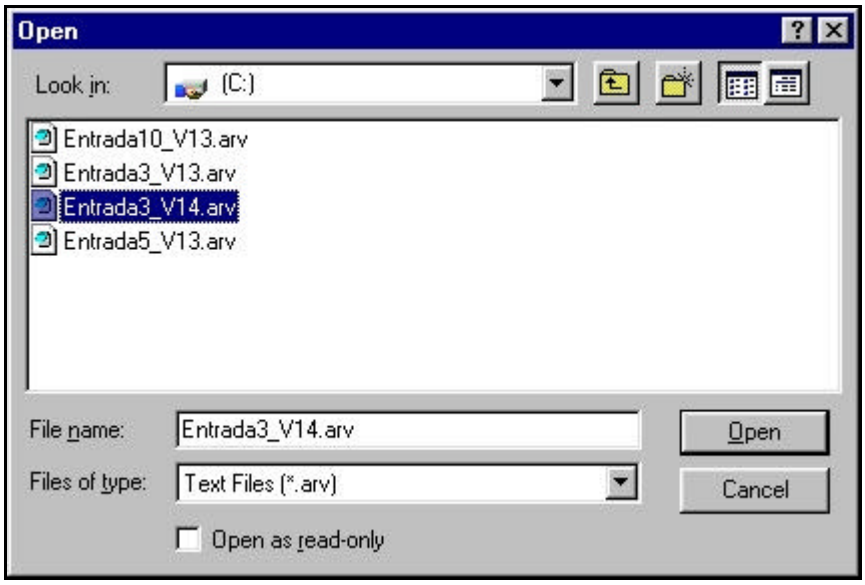


Como pode ser observado, a barra de botões também pode ser removida a critério do usuário que estiver operando o sistema. Caso o usuário queira ver novamente a barra de botões, ele deve ativar a opção “Parâmetros” no menu “Executar” e marcar esta opção novamente.

Os detalhes desta tela encontram-se na seção A.11.3 Tela de Comprimentos deste manual.

A.12.4. Ler Arquivo de Dados

A seguir, o sistema solicita o nome do arquivo de dados para a busca.



Esta é uma tela padrão do Windows e não requer maiores explicações. Ela permite navegar pelo sistema de arquivos da máquina do usuário para a seleção do arquivo desejado.

Se o usuário clicar no botão "*Cancel*" o sistema pára a busca e aguarda o próximo comando.

Para prosseguir deve-se selecionar o arquivo de dados e clicar no botão "*Open*".

A.12.5. Examinar os Resultados

Dependendo das opções selecionadas, os resultados serão apresentados somente na tela principal ou também na tela de "Comprimentos".

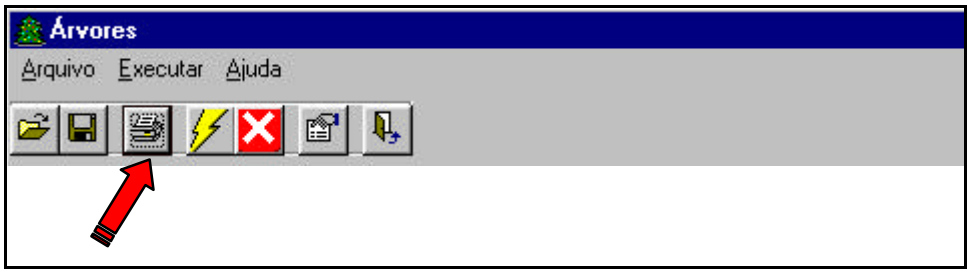
Se não for selecionado o método de cálculo "Comparativo" na tela de parâmetros, os resultados serão mostrados somente na tela principal, sendo que a aba "*Likelihood Ratio Test* (*LRT*)" desta tela e a tela "Comprimentos" ficarão em branco.

Se for selecionado o método de cálculo "Comparativo" na tela de parâmetros, todas as telas de resultados serão preenchidas com informações relevantes para a pesquisa.

Maiores detalhes sobre os resultados podem ser obtidos na seção A.11. Telas do Sistema deste manual.

A.12.6. Imprimir Uma Árvore

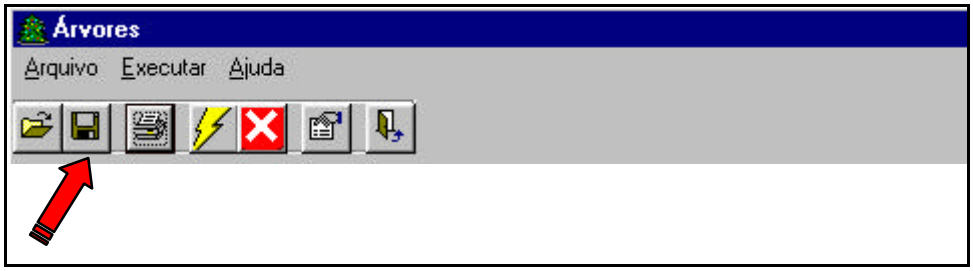
O sistema permite impressão de uma árvore. Será impressa a árvore que estiver ativa na aba “Árvores” da tela principal do sistema.



Esta opção do sistema também pode ser ativada pela opção “Imprimir” do menu “Arquivo”.

A.12.7. Gravar Arquivo Com os Dados

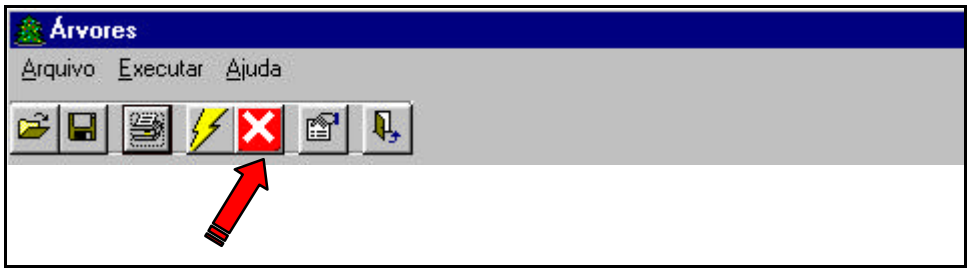
O sistema permite que sejam gravados em arquivo todos os dados utilizados até o momento na pesquisa. Todos os elementos e todas as opções definidas na tela de parâmetros serão armazenados para futura consulta ou reinício da pesquisa.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Gravar arquivo” no menu “Arquivo”.

A.12.8. Parar a Busca

O processo de busca das melhores árvores pode demorar e alguns ajustes nos parâmetros podem ajudar a melhorar a performance do sistema. Para interromper o processamento foi criado o botão assinalado na figura a seguir. Após o encerramento o processo pode ser reiniciado novamente sem prejuízo das informações encontradas até o momento.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Parar busca” no menu “Executar”.

A.12.9. Reiniciar a Busca

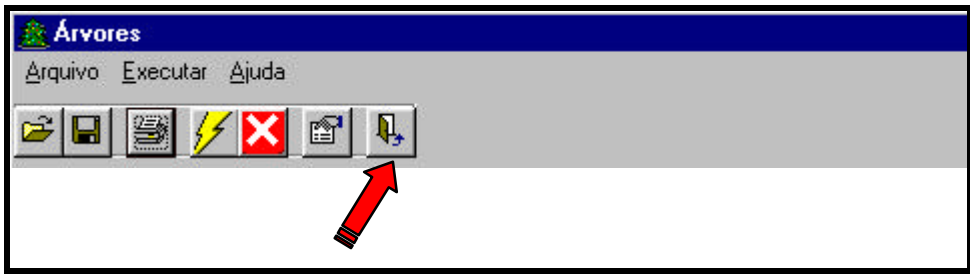
O reinício e início são ativados pelo mesmo botão, assinalado na figura a seguir.

O sistema permite que a busca seja reiniciada após sua interrupção ou finalização. Esta funcionalidade é particularmente útil quando se deseja estudar o impacto causado pela alteração de algum parâmetro ou quando se deseja examinar um novo conjunto de elementos.

Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Iniciar” no menu “Executar”.

A.12.10. Encerrar Sistema

Para encerrar o sistema foi criado o botão assinalado na figura a seguir. O sistema ainda fará mais uma pergunta pedindo confirmação do usuário antes de tirá-lo do ar.



Esta funcionalidade pode ser ativada a partir do botão assinalado na figura anterior e também pela opção “Sair” no menu “Arquivo”.

A.13. Configuração do Arquivo com as Seqüências de DNA

Os tópicos obrigatórios são somente os dois primeiros do arquivo: [Cabec] e [DNA].

Quando não for possível determinar algum dos parâmetros do sistema deve-se informar apenas até o tópico [Parametros] (exclusive). A partir deste ponto, o sistema assume os valores *default* e permite que o usuário os altere quando necessário.

```
[Cabec]
Projeto Árvores Filogenéticas
Versao: 1.4 - Demonstração
Data: 15/jan/2001
Autor: Oclair Prado
Seqüência em análise: DNA de Mitocôndrias
Elementos: 3
[DNA]
Sequencia_1
GTAAATGACA
Sequencia_2
ATAAATGACA
Sequencia_3
CTAAACAACA
[Parametros]
Metodo: 3
Alfa: .6
Beta: .3
```

```
Pop. total: 3
Pop. interm.: 3
Limite de iterac.: 100
Melhores elementos: 40
Piores elementos: 10
Mostrar param.: 0
Mostrar bot.: 1
Estrat. busca.: 2
Fit. ideal: .1
Troca folha RW: 20
Troca folha ME: 30
Troca arv. RW: 20
Troca arv. ME: 30
Limite horas: 10
Limite minutos: 0
[Populacao]
4*3*.232334734391942*4*5*.264756190936621*5*1*8.26909161713267E-
02*5*2*9.21986494892598E-02*
4*3*0*4*5*0*5*1*0*5*2*0*
4*3*0*4*5*0*5*1*0*5*2*0*
```

A população é gravada usando o conceito de matriz de adjacências. Cada elemento é representado internamente por um grafo e as coordenadas de sua matriz de adjacências são gravadas nesta parte do arquivo. Eles são gravados em conjuntos de três informações, separadas por um asterisco, sendo que a primeira corresponde ao nó pai, a segunda corresponde ao nó filho e a terceira corresponde ao comprimento da aresta que une pai e filho. Por exemplo, o primeiro conjunto, representado por **"4*3*.23233473439142"**, informa ao sistema que o nó **4** é pai do nó **3** e a aresta que os une tem comprimento igual a **0,23233473439142**.

Apêndice B

Cálculo do número de árvores, ramos e nós de uma árvore

B.1 Cálculo do número de árvores, ramos e nós de uma árvore dado o número de folhas

Segundo Nei & Kumar (2000), em geral, o número de possíveis topologias para uma árvore bifurcada com raiz, de tamanho igual a n elementos (ou folhas), é dado por:

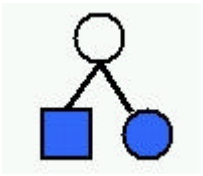
$$1*3*5*...*(2n-3)=\prod_{i=2}^{n}[2(i+1)-5]=\frac{(2n-3)!}{2^{n-2}(n-2)!}$$

para $n \geq 2$ (Cavalli-Sforza & Edwards, 1967). Isto indica que os números de topologias para $n=2$, 3, 4, 5 e 6 são 1, 3, 15, 105 e 945, respectivamente. Quando $n=10$, temos $1*3*5*7*9*11*13*15*17=34.459.425$.

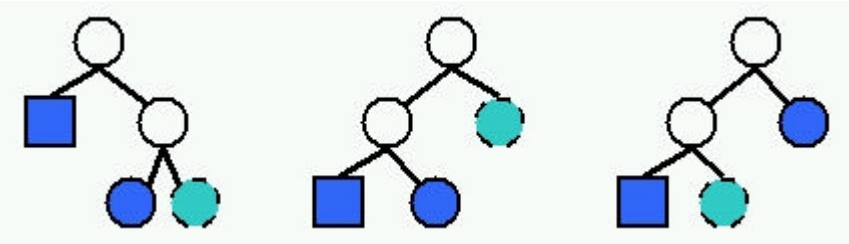
Em uma árvore bifurcada com raiz, o número de ramos internos e o de nós internos são $n-2$ e $n-1$, respectivamente, e o número total de ramos é $2n-2$.

Demonstração

Para duas folhas (n = 2) existe apenas uma topologia possível.



A terceira folha a ser inserida pode se combinar com cada um dos três nós, formando sempre árvores diferentes



A quarta folha a ser inserida pode se combinar com cada um dos cinco nós, e assim sucessivamente, até a enésima folha que poderá se combinar com cada um dos $2n-3$ nós. Este processo pode ser representado pelo produtório que foi simplificado usando fatorial, conforme exibido a seguir.

$$1*3*5*...*(2n-3)=\prod_{i=2}^{n}[2i-3]$$

Multiplicando o numerador e o denominador da expressão anterior por $\prod_{k=2}^{n-1}[2k-2]=2*4*6*8*...*(2(n-2)-2)*(2(n-1)-2)$, teremos:

$$\frac{1*2*3*4*5*6*...*(2(n-1)-2)*(2n-3)}{2*4*6*...*(2(n-2)-2)*(2(n-1)-2)}=\frac{1*2*3*4*5*6*...*(2n-4)*(2n-3)}{2*4*6*...*(2n-6)*(2n-4)}$$

O numerador pode agora ser substituído por $(2n-3)!$.

$$\frac{(2n-3)!}{2*4*6*...*(2n-6)*(2n-4)}$$

Como temos $n-2$ fatores no denominador onde aparece o 2, podemos realizar seu isolamento, que passará a valer 2^{n-2}, e a fórmula pode ser escrita da seguinte maneira:

$$\frac{(2n-3)!}{2^{n-2}*[1*2*...*(n-3)*(n-2)]}$$

Finalmente, substituindo no denominador $[1*2*3*...*(n-3)*(n-2)]$ por $(n-2)!$, retornamos à fórmula que gostaríamos de demonstrar:

$$1*3*5*...*(2n-3) = \prod_{i=2}^{n}[2i-3] = \frac{(2n-3)!}{2^{n-2}(n-2)!}$$

Apêndice C

Estruturas de dados

C.1 Introdução

Estruturas de dados são os blocos construtores (*building blocks*) dos algoritmos de computadores (Manber, 1989). O projeto de um algoritmo é parecido com o de um edifício. É necessário colocar todos os aposentos juntos, da maneira mais eficiente possível, de acordo com o propósito do edifício. Para atender neste requisito, não basta somente o conhecimento sobre funcionalidade, eficiência, forma e beleza. É necessário conhecimento completo sobre as técnicas de construção. Projetar um quarto flutuando no ar pode atender ao resultado esperado mas sua implementação na prática é impossível. Outras idéias mirabolantes podem até ser possíveis, porém muito caras. Da mesma forma, o projeto de um algoritmo deve ser baseado em sólida compreensão das técnicas de estruturas de dados e seus custos (Manber, 1989).

Neste apêndice, será realizada uma revisão das estruturas de dados básicas normalmente utilizadas em informática e que serviram de base para o desenvolvimento do **Projeto Árvores Filogenéticas**, utilizado para esta pesquisa.

C.2 Estruturas de dados elementares

C.2.1 Elemento

A noção de elemento neste apêndice é a de um nome genérico para um tipo de dado não específico. Um elemento pode ser um inteiro, um conjunto de inteiros, um arquivo no formato texto ou outra estrutura de dados. Será usada a noção de elemento em todas os lugares onde o tipo de dado for irrelevante (Manber, 1989).

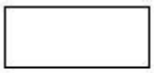


Figura C.1 Representação de um elemento abstrato (Manber, 1989).

C.2.2 Vetor

Um vetor é uma linha de elementos do mesmo tipo. O tamanho do vetor é igual ao número de seus elementos e deve ser fixo. A área de memória alocada para um vetor é sempre consecutiva. Se o primeiro byte de um vetor estiver alocado na posição X da memória, então o *K-ésimo* byte deste vetor estará alocado na posição X + K - 1. Conseqüentemente, é fácil calcular a posição inicial de cada elemento de um vetor (Manber, 1989).

De acordo com Manber (1989), vetores são muito eficientes e muito comuns. Todo elemento de um vetor apresenta tempo de acesso constante, independente de ser o primeiro ou o último. As principais desvantagens do uso de vetores são:

(a) não é possível armazenar elementos de tipos diferentes em um mesmo vetor e

(b) o tamanho de um vetor não pode ser alterado dinamicamente.

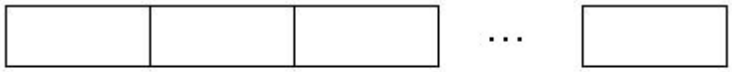


Figura C.2 Representação de um vetor. Todos os seus elementos são do mesmo tipo (Manber, 1989).

Ao contrário de vetor, a lista pode ter seu tamanho alterado em tempo de execução de um programa. Este tipo de estrutura de tamanho variável será analisado no item C.2.4 Lista Ligada.

C.2.3 Registro

Registros são semelhantes aos vetores, exceto por não assumirmos que todos os seus elementos sejam do mesmo tipo. Assim como nos vetores, o tamanho necessário para o armazenamento de um registro é conhecido "*a priori*". Qualquer elemento de um registro pode ser acessado em tempo constante, ou seja, o tempo de acesso é o mesmo para qualquer elemento de um registro. Semelhante aos vetores, os registros também são armazenados em porções consecutivas de memória e também não podem ter seu tamanho alterado de maneira dinâmica (Manber, 1989).

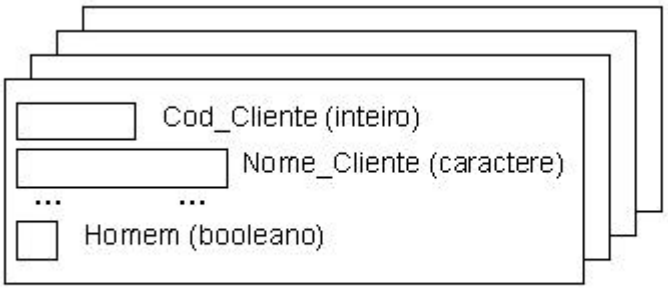


Figura C.3 Representação de um registro. Os seus elementos podem ser de tipos diferentes (Manber, 1989).

C.2.4 Lista ligada

As listas ligadas são as mais simples formas de estruturas de dados dinâmicas. Existem muitas aplicações em que o número de elementos de uma lista deve ser alterado dinamicamente. É possível definir todos os elementos como sendo vetores (ou registros) grandes o bastante para assegurar espaço de memória suficiente para todos os elementos a serem usados. Esta é freqüentemente uma boa solução, mas, é claro, isto não é muito eficiente por necessitar de espaço em memória de acordo com o pior caso (e, em muitos casos, o pior caso nem é conhecido). Complementando, existem casos em que é necessário inserir ou remover um novo elemento no meio da lista. Se estivermos usando vetores e for necessário inserir um novo elemento no meio, teremos que mover todos os outros elementos (Manber, 1989).

Para inserir novos elementos ou remover os já existentes é necessário abandonar a representação consecutiva dos vetores. Ao contrário, cada elemento deve ser representado separadamente, e todos os elementos são conectados através do uso de apontadores (ou ponteiros). Um apontador é simplesmente uma variável que contém o endereço de outro elemento. Uma lista ligada é uma lista de pares, sendo que cada um é composto por um elemento e um apontador, de modo que cada apontador contenha o endereço do próximo par. Cada um destes pares é representado por um registro. Uma lista ligada pode ser percorrida seguindo os endereços dos apontadores. Este percurso deverá ser sempre linear. Isto significa que não é possível acessar diretamente cada um dos elementos, é necessário percorrer a lista ordenadamente iniciando pelo primeiro elemento até chegar no elemento desejado (Manber, 1989).

Segundo Manber (1989), existem duas grandes desvantagens com esta representação de lista ligada:

(a) Ela requer mais espaço por necessitar de um apontador para cada elemento da lista e

(b) Se desejarmos verificar o conteúdo do trigésimo elemento, deveremos iniciar a busca pelo início da lista e passar pelos outros 29 elementos anteriores, um de cada vez.

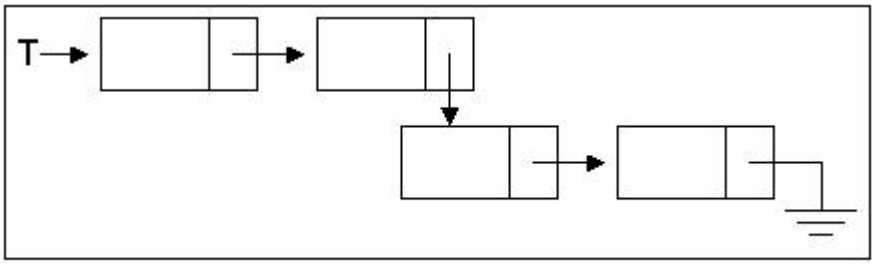


Figura C.4 Representação de uma lista ligada (Manber, 1989).

C.3 Pilha

Conforme pode ser verificado em Horowitz & Sahni (1984), pilha (*stack*) é uma das estruturas de dados mais comuns nos algoritmos de computação.

Pilha é uma lista ordenada na qual todas as inserções e retiradas são feitas numa extremidade, chamada topo (Horowitz & Sahni, 1984).

A restrição de uma estrutura de dados do tipo pilha implica que os seus elementos A, B, C, D e E são acrescentados nesta ordem à pilha e então o primeiro elemento a ser retirado/suprimido deve ser E. Em outras palavras, podemos dizer que o último elemento a ser inserido na pilha será o primeiro a ser retirado. Por esta razão, as pilhas também podem ser referenciadas como listas *Last In First Out* (**LIFO**) (Horowitz & Sahni, 1984).

Um exemplo natural do uso de pilha que surge na programação de computador é o processamento de chamadas de sub-rotinas e seus retornos. Quando o processo principal executa uma rotina, ele primeiro empilha o endereço de retorno para que o processador possa saber para onde deverá devolver o controle de execução após terminar a execução da rotina invocada (Horowitz & Sahni, 1984). Esta implementação usando pilha permite o uso de subrotinas recursivas.

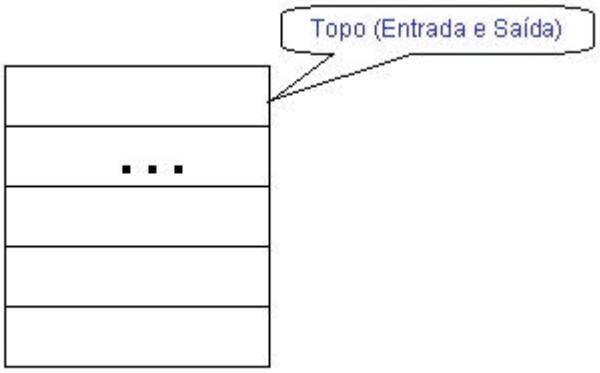


Figura C.5 Representação de uma pilha (Horowitz & Sahni, 1984).

C.4 Fila

Conforme descrito por Horowitz & Sahni (1984), fila (*queue*), assim como pilha, é um dos tipos mais comuns de estruturas de dados usados em algoritmos de computação.

Fila é uma lista ordenada na qual todas as inserções ocorrem numa extremidade, a frontal, enquanto que todas as retiradas ocorrem na outra extremidade, a traseira (Horowitz & Sahni, 1984).

As restrições sobre uma estrutura de dados do tipo fila exigem que o primeiro elemento inserido na fila seja o primeiro a ser removido. Assim, as filas são conhecidas como listas *Fist In Last Out* (**FILO**) (Horowitz & Sahni, 1984).

Um exemplo muito comum do uso de estruturas de dados do tipo fila é o de planejar serviços. Em processamento de lotes, os serviços são “enfileirados” na medida em que são preparados na entrada, e executados um após o outro, na mesma ordem em que foram recebidos (Horowitz & Sahni, 1984).

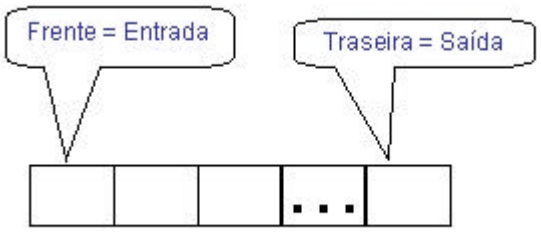


Figura C.6 Representação de uma fila (Horowitz & Sahni, 1984).

C.5 Árvores

A única forma de organização que vetores e listas ligadas são capazes de representar é a ordenação de seus elementos. Existem muitas aplicações que requerem outras formas de organização. Árvores também são capazes de representar hierarquia. Elas também servem como uma estrutura de dados mais eficiente para certas operações que requerem estruturas lineares (Manber, 1989).

Uma árvore com raiz é um conjunto de elementos chamados nós (ou vértices) junto com um conjunto de arestas que conectam os elementos de uma forma especial (veja Figura C.7). Um nó é a raiz (o topo da hierarquia). A raiz é conectada aos outros nós, que representam o nível 1 da hierarquia; eles, por sua vez, são conectados aos nós no nível 2, e assim por diante. Todas as conexões são entre nós e seus superiores diretos (normalmente chamados pais). Somente a raiz não possui pai. A principal propriedade das árvores é que elas não possuem ciclos. Como um resultado, existe somente um caminho entre qualquer par de nós de uma árvore (Manber, 1989).

Um nó é conectado ao seu pai e a alguns filhos. O número máximo de filhos de qualquer nó é chamado de grau da árvore. O caso especial de árvore de grau 2 é chamado de **árvore binária** e identificamos os filhos por “esquerdo” e “direito”. Um nó sem filhos é chamado de folha. Um nó que não é folha é chamado simplesmente de nó interior. A altura de uma árvore é igual ao seu nível máximo, também chamado de distância máxima entre a raiz e as folhas (Manber, 1989).

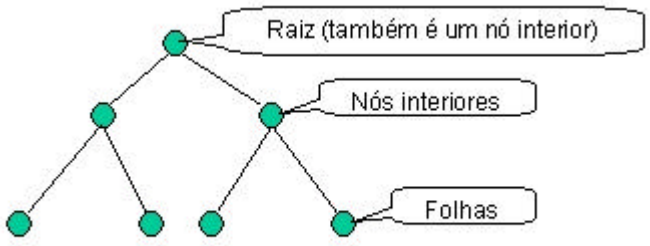


Figura C.7 Representação de uma árvore (Manber, 1989).

C.6 Grafos

Um grafo $G = (V, E)$ consiste em um conjunto V de **nós** (também chamados vértices) e de um conjunto E de **arestas**. Cada aresta corresponde a um par de nós. As arestas representam relacionamentos entre os nós. Por exemplo, um grafo pode representar um

conjunto de pessoas e as arestas podem conectar duas pessoas que se conheçam. Um grafo pode ser dirigido ou não dirigido. As arestas em um grafo dirigido são pares ordenados pois a ordem entre dois nós conectados pela aresta é importante. Neste caso representamos a aresta como sendo uma seta apontando de um nó (cauda) para outro (cabeça). Os nós em um grafo não dirigido são pares desordenados. Árvores são exemplos de grafos com um caminho único entre os nós. A Figura C.8 apresenta as principais representações gráficas de grafos. A primeira representação é a **matriz de adjacências** de um grafo (Figura C.8 (a)). Seja $|V|=n$, a matriz de adjacências de G é uma matriz A_{nxn} tal que $a_{ij}=1$ se e somente se $(v_i,v_j)\in E$, caso contrário $a_{ij}=0$. A maior desvantagem da matriz de adjacências é o espaço de memória necessário para sua representação. Será sempre n^2, independente do número de vértices ou de arestas. Por exemplo, o número de arestas em uma árvore é $n-1$ e estas arestas podem ser representadas por um ou dois apontadores por vértice. Usando matriz de adjacências, cada vértice possui um vetor associado de tamanho n. Se o número de arestas for pequeno, a maioria dos elementos da matriz de adjacências será zero (Manber, 1989).

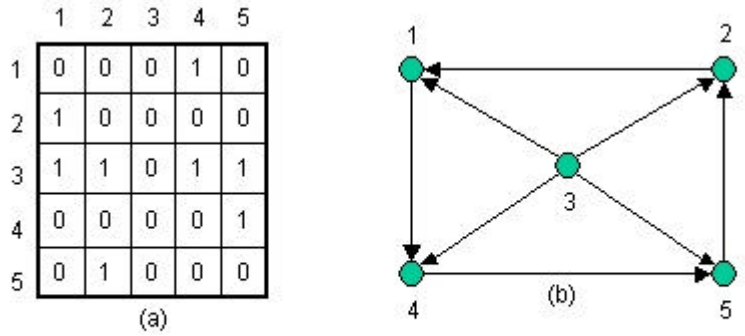


Figura C.8 Representações de um grafo, onde: (a) é uma matriz de adjacências e (b) é sua representação gráfica mais comum (Manber, 1989).

O grafo representado na Figura 8 (b), é do tipo dirigido, pois suas arestas indicam uma direção única entre os nós. Por exemplo, é possível caminhar do nó 2 até o nó 1, mas não é possível ir direto do nó 1 para o nó 2.

C.7 Conclusões

As estruturas de dados podem ser divididas em estáticas e dinâmicas. Vetores são estruturas estáticas. O tamanho de um vetor deve ser conhecido antes de iniciarmos seu uso e ele não pode ser estendido. Por outro lado, acessar um vetor é muito eficiente. Listas ligadas

são estruturas dinâmicas. Elas podem facilmente ter seu tamanho estendido ou reduzido. Elas podem ter qualquer tamanho dentro dos limites do sistema operacional e do total de memória disponível (Manber, 1989).

Estruturas de dados podem também ser divididas em unidimensionais e multidimensionais. Vetores e listas ligadas são unidimensionais. A única forma de organização que eles podem representar é uma possível ordenação entre seus elementos. Árvores representam um pouco mais do que somente uma forma de organização, elas podem representar hierarquia. Grafos podem representar formas de ordenação ainda mais elaboradas do que as árvores (Manber, 1989).