UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA
PROGRAMA DE PÓS-GRADUAÇÃO EM BIOTECNOLOGIA

MARCELO VERA CRUZ DINIZ

ANÁLISE COMPUTACIONAL DE SINTASES DA QUITINA DE FUNGOS BASIDIOMICETOS

Feira de Santana, BA
Maio. 2010

MARCELO VERA CRUZ DINIZ

ANÁLISE COMPUTACIONAL DE SINTASES DA QUITINA DE FUNGOS BASIDIOMICETOS

Dissertação apresentada ao Programa de Pós-graduação em Biotecnologia, da Universidade Estadual de Feira de Santana como requisito parcial para obtenção do título de Mestre em Biotecnologia.

Orientador: Prof. Dr. Aristóteles Góes-Neto

Feira de Santana, BA
Maio. 2010

Ao meu amor,

Silvana Neves Diniz.

AGRADECIMENTOS

Aos meus pais Luciano e Isabela, meu irmão Rodrigo. Devo tudo que sou a vocês. Aos meus sogros Itamar e Dinha e meu cunhado Anderson, vocês são grandes exemplos de vida. Agradeço, principalmente, a minha esposa pela sua compreensão, paciência e tolerância. Amo todos vocês.

Ao meu orientador Aritóteles Goes-Neto, pela oportunidade de trabalho, imensa capacidade criativa, disposição e motivação no trabalho.

Aos meus grandes amigos Bruno Silva Andrade e Catiane Sacramento Souza pelas horas de sugestões, ensinamentos e risadas.

Aos professores Roberto Andrade, Thierry Petit, Charbel El-Hani, a professora Suani Pinho, e aos meus colegas Leonardo Bacelar e Ivan Rocha pelo apoio e aprendizado. Quero sempre fazer parte desse grupo fantástico, o FESC.

Ao programa de Pós Graduação em Biotecnologia - PPGBiotec, pela disponibilização da ótima estrutura física e humana.

A Deus por sempre me iluminar, proteger e mostrar os caminhos da verdade.

AGRADECIMENTO ESPECIAL

Pelo incentivo e envolvimento em cada passo que dei, agradeço ao prof. Dr. Aristóteles Góes-Neto pela confiança que me deu, deixando-me caminhar com as próprias pernas; não teria chegado até aqui sem sua excelente orientação. Muito obrigado.

“A cada nova existência, o homem tem mais inteligência e pode melhor distinguir o bem e o mal”.

Allan Kardec.

SUMÁRIO

RESUMO

Um dos principais componentes da parede celular fúngica é o polissacarídeo quitina que é sintetizado a partir da atividade da enzima sintase da quitina. Este projeto teve como objetivo realizar um estudo integrado in silico de sequências aminoacídicas de sintases da quitina de fungos basidiomicetos através de métodos de análise comparativa de perfis e padrões, análise filogenética e construção de redes complexas. Os seguintes produtos serão gerados ao final do projeto: (I) um banco de dados relacional contendo todas as sequências protéicas, completas ou parciais, de sintases da quitina de Basidiomycota construído e validado; (II) identificação e caracterização dos domínios conservados das sequências protéicas; (III) árvores filogenéticas das sequências protéicas; e (IV) redes complexas das sequências protéicas de sintases da quitina de Basidiomycota. Os resultados subsidiarão o desenvolvimento, através de modelagem molecular, de novos compostos ou de modificações químicas de compostos pré-existentes, direcionados à inibição da síntese da quitina na parede celular de fungos basidiomicetos fitopatógenos.

Palavras-chave: *sintase da quitina, Basidiomycota, filogenia, redes complexas, biotecnologia.*

ABSTRACT

One of the main components of the fungal cell wall is the polysaccharide chitin, which is synthesized by the enzyme chitin synthase. Our work aimed to carry out an integrated in silico study of protein sequences of basidiomycotan chitin synthases using methods of comparative analysis of profiles and patterns, phylogenetic analyses, and complex network approach. Initially, we construct a relational database The following products were generated at the end of the project: (I) a relational database containing all complete or partial protein sequences of chitin syntases of Basidiomycota, constructed and validated; (II) identification and characterization of conserved domains of protein sequences; (III) phylogenetic trees of protein sequences; and (IV) complex networks protein sequences of basidiomycotan chitin synthases. The results will subsidize the development, by molecular modelling, of new compounds or chemical modifications of pré-existent compounds, addressed to the inhibition of the chitin synthesis in the cell wall of phytopathogenic Basidiomycota.

Keywords: *chitin synthase, Basidiomycota, phylogeny, complex networks, biotechnology.*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

% - Percentagem
ANSI - *American National Standards Institute*
API - *Application Programming Interface*
BLAST - *Basic Local Alignment Search Tool*
BLOSUM - *Blocks substitution matrix*
FESC – Grupo de Física e Estatística Computacional da UFBA
GB - *Gigabyte*
Ghz - *Gigahertz*
HTTP - *Hypertext Transfer Protocol*
NCBI - *National Center for Biotechnology Information*
PAM – *Point Accepted Mutation*
PC – *Personal Computer*
PERL - *Pratical Extraction and Reporting Language*
RAM – *Random access memory*
SGBD - Sistema Gerenciador de Banco de Dados
SQL - *Structured Query Language*
UEFS - Universidade Estadual de Feira de Santana
UFBA – Universidade Federal do Estado da Bahia

1. INTRODUÇÃO

A bioinformática se desenvolveu a partir do final dos anos 80, do século XX, principalmente devido à enorme massa de dados gerada pelos grandes projetos na área de genômica (POLANSKI; KIMMEL, 2007). No seu sentido mais amplo, o termo bioinformática compreende a aplicação da tecnologia da informação para o armazenamento, gerenciamento, análise e comunicação de dados biológicos. Suas inúmeras ferramentas analíticas permitem a mineração de dados nessas extensas bases de dados.

A análise de sequências aminoacídicas (proteínas) utilizando métodos tradicionais de filogenia é a opção mais adotada pela comunidade acadêmica. Porém, os métodos mais utilizados (inferência Bayesiana e a análise de Verossimilhança) utilizam modelos matemáticos computacionalmente custosos, e devido a isso, a utilização de um grande número de sequências torna-se inviável. Em contra partida, a comparação por similaridade e a análise de padrões, aliados as Redes Complexas, possibilitam estudos integrados das sequencias sob abordagens metodológicas diferentes. Dessa forma, é possível faz um estudo em larga escala, ou seja, utilizando um grande número de sequências, sem fazer grandes investimentos na plataforma computacional.

Atualmente, a taxonomia de Basidiomycota está baseada em análises filogenéticas de sequências nucleotídicas de genes que codificam RNA ribossômicos (nucleares e mitocondriais), seus espaçadores (seguimentos intergêncios que estão entre os genes de RNA ribossômico) e alguns genes codificadores de proteínas nucleares (subunidades da RNA polimerase II e fator de alongamento I) e mitocondriais (ATP6) (JAMES et al., 2006; HIBBETT et al., 2007). Entretanto, nenhum desses genes codificadores de proteínas, utilizados na reconstrução filogenética, está ligado a rotas metabólicas exclusivas de fungos, assim como não há registro de trabalhos envolvendo teoria de Redes Complexas sobre o tema.

Como a parede celular é geralmente o alvo de compostos antifúngicos atualmente existentes e, uma vez que a quitina é exclusivamente encontrada em fungos e não em plantas, um estudo integrado *in silico* das proteínas enzimáticas

sintases da quitina dos Basidiomycota subsidiará estudos futuros de inibidores desta e, por conseguinte, da inibição da síntese de quitina em basidiomicetos patogênicos.

Este projeto tem como objetivo geral conduzir uma análise computacional em larga escala das sequências protéicas de sintases da quitina de Basidiomycota. Os objetivos específicos consistem na (i) construção de um banco de dados relacional contendo todas as sequências protéicas, completas e parciais, de sintases da quitina de Basidiomycota armazenadas no NCBI (NCBI, 2007) até o dia 11/09/2009; (ii) construção e análise das redes complexas das sequências protéicas de sintases da quitina de Basidiomycota; (iii) análises filogenéticas destas sequências, utilizando as técnicas de distância, inferência bayesiana, parcimônia e verossimilhança, (iv) comparação entre os resultados obtidos entre as análises filogenéticas e a análise de redes complexas destas sequências e (v) identificação, caracterização e análise dos domínios conservados nestas sequências para a sugestão de alvos moleculares genéricos e específicos contra as sintases de quitina de Basidiomycota.

.

2. PAREDE CELULAR DE FUNGOS BASIDIOMICETOS

A parede celular é uma estrutura externa à membrana plasmática da qual depende a vida da hifa ou célula fúngica. É um arcabouço que sustenta a célula do fungo e ao mesmo tempo interage com o ambiente. Sua integridade é essencial à sobrevivência de suas hifas em ambientes hostis, e está presente nas hifas (fungos filamentosos) ou células (leveduras) dos diferentes grupos fúngicos (RONCERO, 2002). Ela protege a célula fúngica contra variações osmóticas, químicas e biológicas, e está envolvida em várias outras funções incluindo morfogênese, expressão antigênica, adesão e interação célula-célula, e ainda desempenha papel fundamental no crescimento, desenvolvimento e interações dos fungos com o ambiente e com outras células (BOWMAN; FREE, 2006).

A parede celular é uma estrutura onde a arquitetura e composição é regulada de forma coordenada com o crescimento da célula, tendo polissacarídeos (quitina, glicanos e mananos) e glicoproteínas como seus principais componentes (BOWMAN; FREE, 2006). É altamente dinâmica e está sujeita às constantes mudanças, como, por exemplo, durante a expansão e divisão celular nas leveduras, e durante a germinação de esporos e formação de septos e crescimento apical de hifas em fungos filamentosos (BOWMAN; FREE, 2006).

A osmose é um processo físico no qual a água se movimenta entre dois meios com diferentes concentração de soluto separados por uma membrana semi-permeável. Esse processo regula a concentração de soluto, definindo características hipotônicas, menor concentração de soluto, ou hipertônicas, maior concentração de soluto, ao meio (AMABIS, MARTHO, 2004).

Células fúngicas destituídas de parede celular só podem sobreviver em condições de laboratório, onde o suporte osmótico previne o seu rompimento (RONCERO, 2002). A seleção da parede celular como alvo na busca de uma defesa efetiva justifica-se por ela ser essencial aos fungos, uma vez que não está presente em vertebrados e plantas, de modo que as rotas biossintéticas das moléculas que compõem a parede celular, como a quitina, são importantes alvos

para o desenvolvimento de agentes inibidores do crescimento destes patógenos (GEORGOPAPADAKOUS; TKACZ, 1995; RONCERO, 2002, BOWMAN; FREE, 2006).

A quitina, homopolímero linear de -1,4-N-acetilglucosamina, é um carboidrato estrutural endógeno e um dos principais componentes da parede celular fúngica. Em fungos, a quitina é sintetizada por uma sequência de cinco reações sucessivas: (i) conversão de Fru-6-P em GlcN-6-P por glutamina-Fru-6-P amidotransferase (E.C. 2.6.1.16), (ii) acetilação de GlcN-6-P gerando GlcNAc-6-P por GlcN-fosfato acetiltransferase (E.C. 2.3.1.4), (iii) interconversão de GlcNAc-6-P em GlcNAc-1-P por N-acetilglicosamina fosfato mutase (E.C. 5.4.2.3), (iv) uridinação de GlcNAc-1-P por UDP- GlcNAc pirofosforilase (E.C. 2.7.7.23) e (v) conversão de UDP-GlcNAc (figura 01) no polímero quitina pela sintase da quitina (E.C. 2.1.4.16) (MIO et al., 1998, LAGORCE et al., 2002).

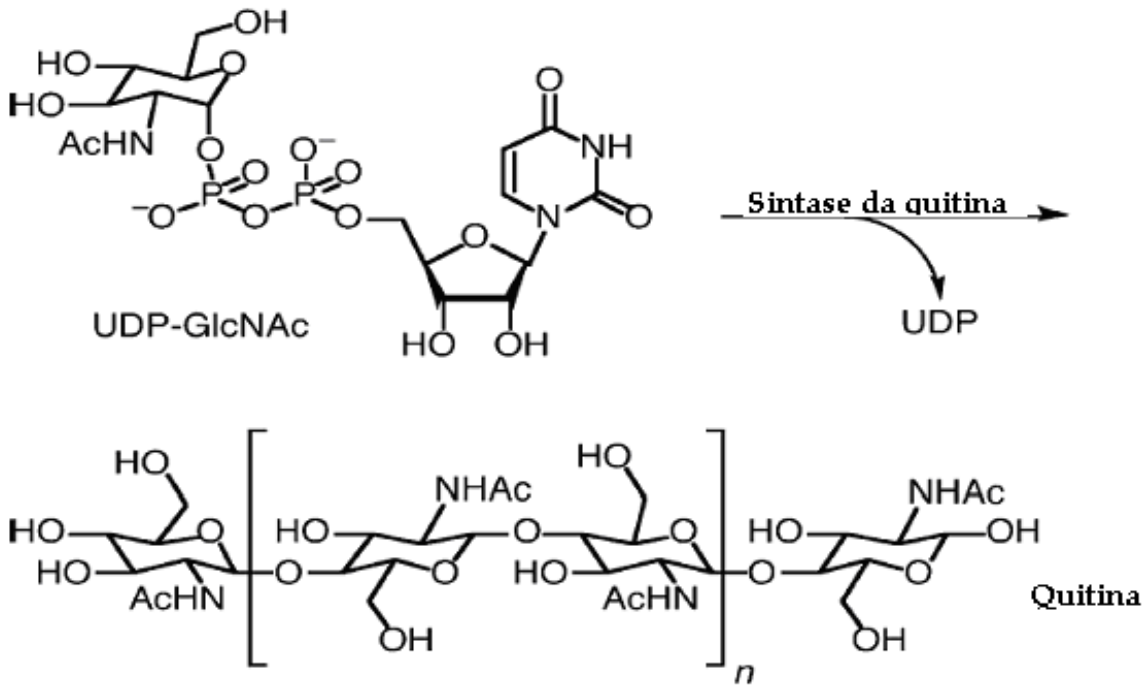


Figura 01: Polimerização da UDP-GlcNAc pela sintase da quitina para formar a quitina (Fonte: YEAGER, FINNEY, 2004).

A quitina é de extrema importância para arquitetura e integridade da parede celular fúngica, uma vez que quando a síntese de quitina é interrompida, a parede

celular se desorganiza e a célula ou hifa fúngica apresenta malformações, tornando-se osmoticamente instável e, geralmente, levando à morte celular (BAGO et al., 1996; SPECHT et al., 1996).

Em nível celular, a quitina é o resultado da atividade da enzima sintase da quitina (CHS), uma glicosiltransferase que converte UDP-N-acetil-D-glucosamina em quitina, originalmente descrita por Glaser e Brown (1957). O primeiro gene codificador de uma sintase da quitina (FKV) fúngica somente foi isolado e caracterizado aproximadamente trinta anos após a descrição da atividade da enzima (BULAWA et al., 1986).

Atualmente, estão depositadas mais de 3000 sequências aminoacídicas, parciais e completas, de sintase de quitina de fungos (NCBI, 2007), entretanto, excetuando o estudo mais genérico de Ruiz-Herrera, González-Prieto, e Ruiz-Medrano (2002), não há registro, até o presente momento de nenhum trabalho comparativo de sintases de quitina de Basidiomycota.

2.1. A Sintase da Quitina

A sintase da quitina (E.C. 2.4.1.16) é uma enzima da família glicosiltransferases, conjunto de enzimas que catalisam as reações de grupos glicosil (açucares), desempenham funções importantes em fungos filamentosos que possuem a quitina como principal componente estrutural de sua parede celular (RUIZ-HERRERA et al., 2002). A sintase da quitina (CHS) apresenta-se em cinco classes diferentes, as quais se diferenciam a partir de seus níveis de expressão. Entre as cinco, a CHS classe III se destaca devido a sua grande importância no ciclo de desenvolvimento dos fungos. Ela atua na formação da parede celular, sendo responsável pela síntese de 90% de toda a quitina da célula.

A Quitina é sintetizada através da seguinte sequência de reações sucessivas: (i) conversão de Fru-6-P em GlcN-6-P por glutamina-Fru-6-P amidotransferase (E.C. 2.6.1.16), (ii) acetilação de GlcN-6-P gerando GlcNAc-6-P por GlcN-fosfato acetiltransferase (E.C. 2.3.1.4), (iii) interconversão de GlcNAc-6-P

em GlcNAc-1-P por N-acetilglicosamina fosfato mutase (E.C. 5.4.2.3), (iv) uridinação de GlcNAc-1-P por UDP- GlcNAc pirofosforilase (E.C. 2.7.7.23) e (v) conversão de UDP-GlcNAc no polímero quitina pela sintase da quitina (E.C. 2.1.4.16) (MIO et al., 1998, LAGORCE et al., 2002) (figura 02). Os genes que codificam a sintase da quitina estão presentes em diferentes organismos distribuídos entre fungos, bacterias, plantas e animais (MERZENDORFER, 2006).

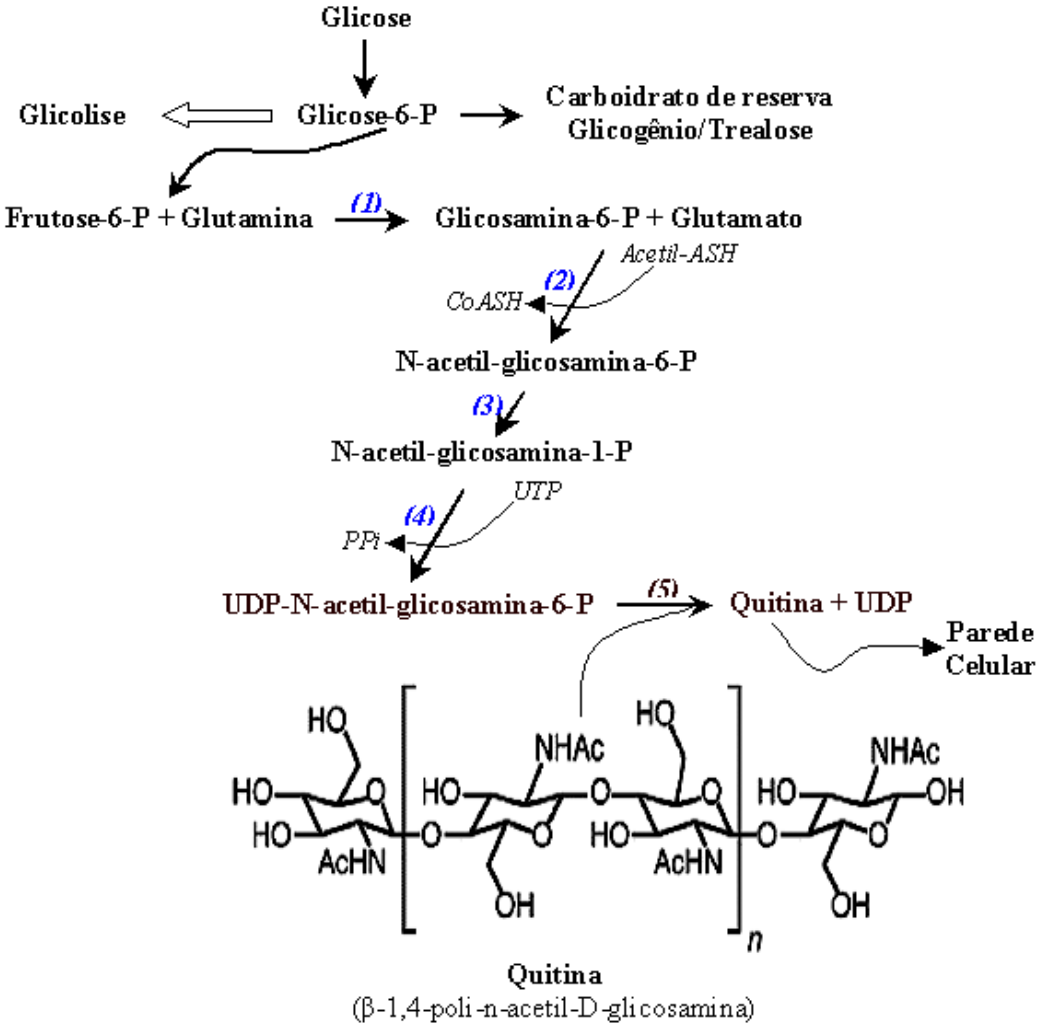


Figura 02: Rota metabólica da sintase da quitina na síntese da parede celular. (1) Glutamina-frutose-6-fosfato amidotransferase (EC 2.6.1.16); (2) Glicosamina fosfato N-acetiltransferase (EC 2.3.1.4); (3) fosfo-N-acetilglicosamina mutase (EC 5.4.2.3); (4) UDP-N-acetilglicosamina pirofosforilase (EC 2.7.7.23) e (5) Sintase da quitina (EC 2.4.1.16) (Fonte: LAGORCE et al., 2002; HOGENKAMP, 2006).

Células de leveduras com vários defeitos, incluindo mutações em alguns genes mostram significativo aumento na sintase da quitina, acompanhado por um aumento na síntese de várias proteínas de parede celular. Esses dados sugerem que estas células reagem contra os danos da parede celular pela ativação de um mecanismo compensatório que garante a sua estabilidade (GARCÍA-RODRIGUEZ, et al., 2000).

3. REDES COMPLEXAS

A teoria das Redes Complexas vem sendo desenvolvida por físicos e matemáticos nas últimas décadas e é considerada uma das teorias mais modernas da ciência contemporânea. Ela é o fruto da união da Teoria dos Grafos e da Mecânica Estatística. As redes de Pequeno Mundo (STROGATZ; WATTS, 1998) e as redes de Livre Escala (BARABÁSI; ALBERT, 1999) são exemplos de Redes Complexas. As redes de Pequeno Mundo modelam redes que não são completamente regulares nem são completamente aleatórias. As Redes de Livre Escala utilizam o conceito de lei de potência para definir a distribuição de graus de uma rede. Essa lei diz que redes que com poucos vértices possuem muitas conexões e redes que com muitos vértices possuem poucas conexões. Essas duas propostas modelam de formas diferentes sistemas que possuem um grande número de arestas e vértices (GALVÃO, 2006).

As Redes Complexas vêm ganhando cada vez mais prestígio e atenção de pesquisadores por ser uma boa ferramenta para modelar sistemas biológicos (NEWMAN, 2007). A sua grande aplicabilidade a tornou uma área multidisciplinar de pesquisa, sendo utilizada para estudar desde a proliferação de células neoplásicas (GALVÃO, 2010), interação entre proteínas (GAVIN, 2004; BARABÁSI, 2004; GÓES-NETO, 2007) e mapeamento de rotas de aeroportos (ROCHA, 2008).

3.1. Teoria dos Grafos

Um grafo G(V,A) é definido pelo conjunto V e A, no qual V é um conjunto, não-vazio, de vértices ou nós e A é um conjunto ordenado de arestas, a(*v*, *w*), onde v e w pertence a V e conectam os nós (NETO, 2006).

Quanto à orientação, um grafo pode ser classificado de duas formas: orientado quando as conexões entre os vértices são orientadas (figura 03a), sendo chamado de dígrafo, ou não-orientado, quando os vértices não possuem

orientação (figura 03b). Deve-se utilizar o termo aresta para definir a conexão entre grafos não-orientados e arcos para a conexão entre grafos orientados.

Cada vértice de um grafo está associado a um conjunto de arestas ou arcos. O grau de um vértice é a propriedade que quantifica essa característica e ela é definida como a quantidade de arestas ou arcos que estão ligados a um vértice. Quando todos os vértices de um grafo têm o mesmo grau diz-se que o grafo é regular (figura 04). A adjacência entre dois vértices é concretizada pela existência de uma aresta ou arco entre eles. (WATTS, 1999).

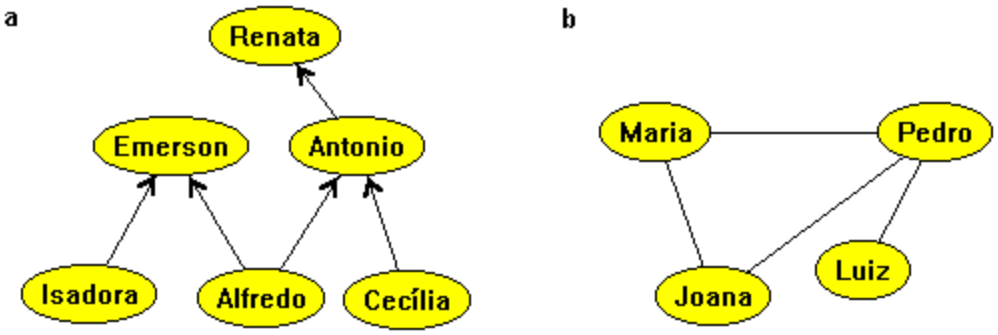


Figura 03: a) Exemplo de Grafo orientado ou Dígrafo G(V, A) onde V = {Renata, Emerson, Antonio, Isadora, Alfredo, Cecília} e A = {(Antonio, Renata), (Cecília, Antonio), (Alfredo, Antonio), (Alfredo, Emerson), (Isadora, Emerson)} e b) Exemplo de Grafo não-orientado G(V, A) onde V = {Maria, Pedro, Joana, Luiz} e A = {(Maria, Pedro), (Joana, Maria), (Pedro, Luiz), (Joana, Pedro)} (MARIANI, 2008).

O conjunto de arestas e vértices que conectam dois vértices em um grafo é chamado de caminho. Um grafo é classificado como conexo (figura 04) se todos os seus vértices são conectados por um caminho (NETO, 2006). Os grafos podem ser representados por diagramas, nos quais os vértices são representados por pontos e as arestas por linhas que conectam os vértices figura 05. Além disso, os grafos também podem ser representados por matrizes de adjacência.

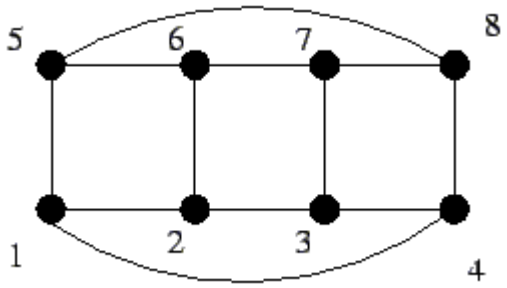
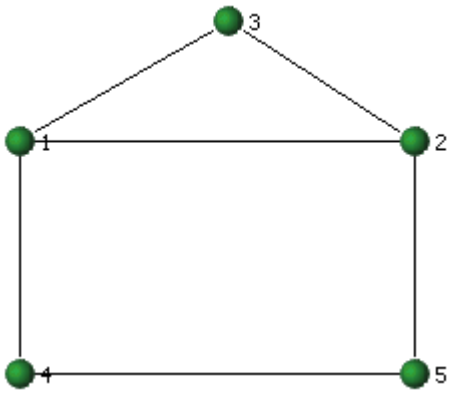


Figura 04: Exemplo de grafo regular e conexo. Todos os vértices têm o mesmo grau e todos os vértices do grafo estão conectados (NONATO, 2010)

Essa representação matemática é construída a partir da seguinte regra: A[*i*, *j*] é definida por $A_{ij} = 1$ se existir uma ligação entre dois vértices, caso não exista o elemento $A_{ij} = 0$ (WATTS, 1999). A figura 05 ilustra um grafo não-orientado com 5 vértices e 6 arestas e a representação da matriz adjacente desse grafo.

a)



b)

	1	2	3	4	5
1	0	1	1	1	0
2	1	0	1	0	1
3	1	1	0	0	0
4	1	0	0	0	1
5	0	1	0	1	0

Figura 05: Ilustração de um grafo e sua matriz de adjacência. a) Ilustração de um grafo não-orientado com 5 vértices e 6 arestas. O conjunto de vértices é B={1,2,3,4,5,6} e o conjunto de arestas é C={ {1,2},{1,3},{1,4}, {2,3}, {2,5}, {5,4}}. b) Representação da matriz de adjacência desse grafo.

Os primeiros estudos que fundamentaram a Teoria dos Grafos sugiram no século XVIII inspirados na antiga cidade prussiana de Konisgsberg, atual Calingrado figura 06, a partir do problema das pontes de Konisgsberg. Esta cidade era cortada por um rio e alguns de seus bairros eram ligados por sete pontes. Os moradores tentavam fazer um passeio passando por cada uma das sete pontes

somente uma vez. Ninguém conseguiu fazer esse trajeto e Leonhard Euler explicou porque isso não era possível.

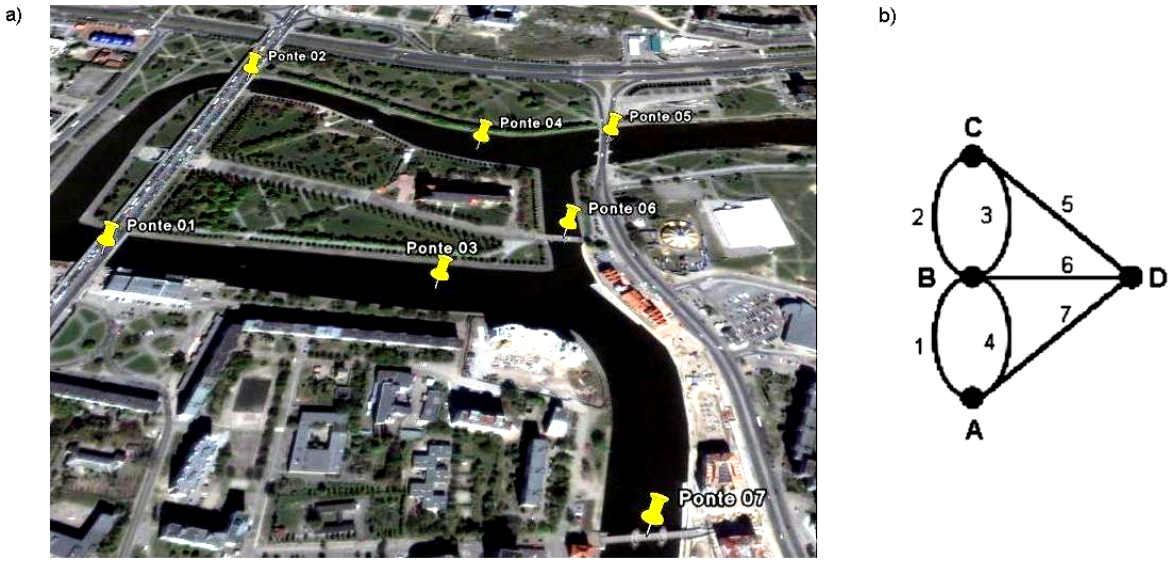


Figura 06: Problema de Konisgsberg. a) Foto atual da cidade de Calingrado, antiga Konisgsberg. b) Representação do diagrama do grafo associado ao problema. Os vértices (A, B, C, D) representam as massas de terra. As arestas (1, 2, 3, 4, 5, 6, 7) representam as sete pontes.

Para que o trajeto tão desejado pelos moradores da antiga Konisgsberg fosse possível o grafo desse problema, representado na figura 06 b, deveria ser conexo e todos os vértices deveriam ter grau par. Além de resolver esse problema, Euler criou uma regra que poderia ser aplicada a qualquer rede de pontes. A topologia de rede que Euler criou é de extrema importância para o estudo das Redes Complexas. Ela, juntamente com a distribuição de grau dos vértices, é responsável pela análise das relações que os Sistemas Complexos modelam (AMARAL, 2004).

3.2. Propriedades das Redes Complexas

Para estudar e poder extrair informações das Redes Complexas, é preciso utilizar parâmetros de medida, os índices da rede. Eles são utilizados para mensurar a grandeza e a complexidade do sistema que está sendo modelado. Existe uma série de índices que podem ser utilizados para estudar Redes Complexas, dentre eles destacam-se o Caminho mínimo médio, Coeficiente de

aglomeração, Distribuição de graus, Assortatividade e *Betweenness*. Essas medidas são as que representam de maneira mais concreta as relações entre os elementos que compõem as Redes Complexas (WANG; CHEN, 2003).

Caminho é o percurso formado pelo conjunto de vértices e arestas que ligam dois vértices. Distância é o conjunto de arestas entre dois vértices, no menor caminho, que os conectam. Com isso, caminho mínimo médio é a média das distâncias entre todos os vértices de um grafo (WANG; CHEN, 2003).

O Coeficiente de Aglomeração de um vértice é a quantidade de arestas que os seus vizinhos têm entre si. Ou seja, esta medida define a probabilidade dos seus vizinhos serem vizinhos entre eles (BARABÁSI, 2002). A equação 1 define o coeficiente de aglomeração de um vértice para um grafo não-orientado:

$$Ci = \frac{2Ei}{Ki(Ki-1)} \qquad (1)$$

Nesta definição, Ei representa número de arestas dos vértices adjacentes ao vértice i e Ki é o número de arestas do vértice i (BARABÁSI, 2002). Para se ter uma visão da rede, analisando essa grandeza, é preciso utilizar o coeficiente de aglomeração médio que é a soma dos coeficientes de aglomeração divididos pela quantidade de vértices. A equação 2 define o coeficiente de aglomeração médio de um grafo:

$$C = \frac{1}{N}\sum_{1}^{N} Ci \qquad (2)$$

O grau de um vértice é dado pelo número de vértices (vizinhos) que ele está conectado (BARABÁSI, 2002). Por exemplo, tenha como base o vértice 1 do grafo representado na Figura 05. O seu grau é 3. Ele está ligado aos vértices 2, 3 e 4. Utilizando a formalidade matemática, diz-se que o grau de um vértice é

|AdjG(i)|, onde i é um determinado vértice do grafo. Partindo desse conceito, o grau médio de um grafo é a media aritmética dos graus de cada vértice. A distribuição de graus é a probabilidade p(k) de um vértice escolhido aleatoriamente ter o grau K (BARABÁSI, 2002).

A Assortatividade mede o grau de similaridade entre os vértices para uma determinada propriedade. Esse índice assume valores entre menos um (-1) e um (+1) e é quantificado devido à semelhança entre os vértices. Quando os vértices são “mais semelhantes entre si” este índice assume valores maiores que zero (0), caso contrário, os valores da assortatividade são menores que zero (0). Normalmente, a propriedade utilizada para análise de assortatividade entre os vértices são os graus dos vértices. É por esta razão que essa propriedade é indicada para estudar agrupamentos de elementos (BARABÁSI, 2002).

Criado em 2004 por Newman e Girvan, o *edge betweenness* (NEWMAN, GIRVAN; 2004), ou *Betweenness* de aresta, é uma medida utilizada para identificar arestas que conectam comunidades, ou agrupamentos de elementos. *Betweenness* de aresta é a soma das frações dos menores caminhos conectados aos pares de nós que passam através de uma aresta (ANDRADE et al, 2009). Essa propriedade atribui valores altos para arestas que conectam comunidades e penaliza as que conectam vértices de um mesmo subgrafo. Com essa propriedade é possível estudar e visualizar a formação de *clusters* (agrupamentos) a partir da sucessiva remoção de arestas de uma rede. A quebra de ligações força a definição de subgrafos que representam indivíduos com as mesmas características. Os passos realizados pelo algoritmo de *Betweenness* são os seguintes:

1) Cálculo do *betweenness* para todas as arestas da rede
2) Busca e remoção da aresta de maior betweenness
3) Recálculo do *betweenness* para as arestas restantes
4) Retorno ao passo 2, até que todas as arestas tenham sido removidas

A figura 07 mostra um exemplo numérico da definição do grau de *Betweenness* ilustrado em (ANDRADE et al, 2009). É possível observar nessa figura o grau de *Betweenness* de cada aresta a partir do vértice *s* = 1 em relações a todos os outros vértices. O procedimento para definição do grau de *Betweenness* inicia pela definição da matriz *Betweenness* B, cujos elementos são definidos pela matriz de adjacência. Ou seja, inicialmente, a matriz de *Betweenness* B é igual à matriz de adjacência. O cálculo do *Betweenness* começa pelos vértices que estão diretamente conectados, e durante a execução dos outros passos do algoritmo os valores correspondentes aos vértices são adicionados a matriz de *Betweenness* B.

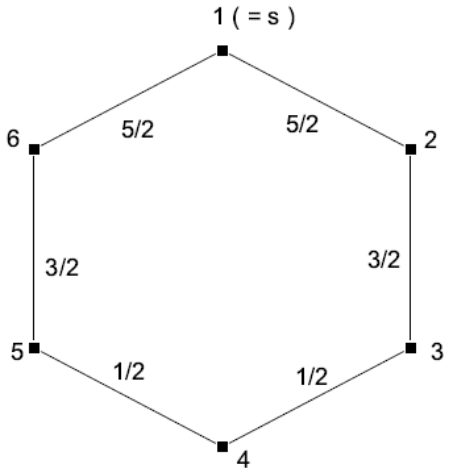


Figura 07: Valor do grau de *Betweenness* de aresta entre os menores caminhos de todos os vértices em relação ao vértice *s* = 1 em uma rede fechada com seis vértices.

O primeiro passo do algoritmo para definição do grau de *Betweenness* executa um laço sobre a variável L que inicia com o valor D, diâmetro da rede, e é decrementado até 2, com $i < j$, nos elementos da matriz de vizinhança D_{ij}. Dentro desse laço são definidos os vértices que estão conectados pela distância L. Esses vértices formam pares (*i*, *j*) que definem as arestas que estão envolvidas nas conexões dos vértices, satisfazendo o requisito 3.

$$d_{i,t(i,j)} = 1 \text{ e } d_{j,t(i,j)} = L - 1 \text{ ou } d_{i,t(i,j)} = 1 \text{ e } d_{j,t(l,j)} = 1 \qquad (3)$$

O segundo passo do algoritmo atualiza a matriz de *Betweenness* B conforme a regra ilustrada em 4. Ele adiciona na matriz B o valor de menor caminho entre os vértices (*i*, *j*). T(*i*, *j*) representa o número de vezes que o requisito (3) foi satisfeito.

$$b_{i,t(i;j)} + (b_{i,j} + 1) = T(i, j) \rightarrow b_{i,t(i,j)} = b_{t(i,j),i} \qquad (4)$$
$$b_{j,t(i;j)} + (b_{i,j} + 1) = T(i, j) \rightarrow b_{j,t(i,j)} = b_{t(i,j),j}$$

O terceiro passo do algoritmo retorna ao primeiro passo até que L seja igual a 2. No final do procedimento a matriz de *Betweenness* B estará completamente atualizada e a posição (*i*, *j*) armazenará o grau de *Betweenness* entre esses dois vértices. Para exemplificar considere as matrizes B, de *Betweenness*, e a matriz D, de distância, ilustradas em 5. Ambas foram construídas a partir da figura 7.

$$D = \begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 3 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 3 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix} \qquad (5)$$

Iniciando o primeiro passo com L = D = 3, encontram-se os valores de d_{ij} = 3, com i < j: (1, 4), (2, 5), (3, 6). Neste ponto é possível observar que o par (1, 4) satisfaz a condição 3 em T(*i*, *j*) = 2, 3, 5, e 6, indicando com T(1, 4) = 4. A atualização descrita em 4 ocorre conforme descrito em 6.

$$b_{1,2} + (b_{1,4} + 1)/4 = 1 + 0 + 1/4 = 5/4 \rightarrow b_{1,2};$$
$$b_{4,2} + (b_{1,4} + 1)/4 = 0 + 0 + 1/4 = 1/4 \rightarrow b_{4,2};$$

$$
\begin{aligned}
&b_{1,3} + (b_{1,4} + 1)/4 = 0 + 0 + 1/4 = 1/4 \rightarrow b_{1,3};\\
&b_{4,3} + (b_{1,4} + 1)/4 = 1 + 0 + 1/4 = 5/4 \rightarrow b_{4,3};\\
&b_{1,5} + (b_{1,4} + 1)/4 = 0 + 0 + 1/4 = 1/4 \rightarrow b_{1,5};\\
&b_{4,5} + (b_{1,4} + 1)/4 = 1 + 0 + 1/4 = 5/4 \rightarrow b_{4,5};\\
&b_{1,6} + (b_{1,4} + 1)/4 = 1 + 0 + 1/4 = 5/4 \rightarrow b_{1,6};\\
&b_{4,6} + (b_{1,4} + 1)/4 = 0 + 0 + 1/4 = 1/4 \rightarrow b_{4,6}:
\end{aligned}
\qquad (6)
$$

O procedimento é exatamente o mesmo para os pares (2,5) e (3,6). Depois de 24 atualizações a matriz de *Betweenness* B alcançaria o estado descrito em 7.

$$
D = \frac{1}{4}\begin{pmatrix}
0 & 6 & 2 & 0 & 2 & 6\\
6 & 0 & 6 & 2 & 0 & 2\\
2 & 6 & 0 & 6 & 2 & 0\\
0 & 2 & 6 & 0 & 6 & 2\\
2 & 0 & 2 & 6 & 0 & 6\\
6 & 2 & 0 & 2 & 6 & 0
\end{pmatrix} \qquad (7)
$$

O ultimo passo, onde L = 2, os valores de (*i*, *j*) onde $d_{ij} = 2$, como $i < j$ são (1, 3), (1, 5), (2, 4), (2, 6), (3, 5), (4, 6). Para cada um desses pares o requisito 3 é satisfeito somente uma vez, logo T(*i*, *j*) = 1. Dessa forma o requisito 3 exige 12 atualizações em elementos da matriz de *Betweenness* B cujo correspondente na matriz de distancia é igual a 1, $d_{ij} = 1$. As atualizações estão descrita em 8.

$$
\begin{aligned}
&b_{1,2} + (b_{1,3} + 1)/1 = 6/4 + 2/4 + 1 = 12/4 \rightarrow b_{1,2};\\
&b_{3,2} + (b_{1,3} + 1)/1 = 6/4 + 2/4 + 1 = 12/4 \rightarrow b_{3,2};\\
&b_{1,6} + (b_{1,5} + 1)/1 = 6/4 + 2/4 + 1 = 12/4 \rightarrow b_{1,6};\\
&b_{5,6} + (b_{1,5} + 1)/1 = 6/4 + 2/4 + 1 = 12/4 \rightarrow b_{5,6};\\
&b_{2,3} + (b_{2,4} + 1)/1 = 12/4 + 2/4 + 1 = 18/4 \rightarrow b_{2,3};\\
&b_{4,3} + (b_{2,4} + 1)/1 = 6/4 + 2/4 + 1 = 12/4 \rightarrow b_{4,3};\\
&b_{2,1} + (b_{2,6} + 1)/1 = 12/4 + 2/4 + 1 = 18/4 \rightarrow b_{2,1};\\
&b_{6,1} + (b_{2,6} + 1)/1 = 12/4 + 2/4 + 1 = 18/4 \rightarrow b_{6,1};\\
&b_{3,4} + (b_{3,5} + 1)/1 = 12/4 + 2/4 + 1 = 18/4 \rightarrow b_{3,4};\\
&b_{5,4} + (b_{3,5} + 1)/1 = 6/4 + 2/4 + 1 = 12/4 \rightarrow b_{5,4};\\
&b_{4,5} + (b_{4,6} + 1)/1 = 12/4 + 2/4 + 1 = 18/4 \rightarrow b_{4,5};\\
&b_{6,5} + (b_{4,6} + 1)/1 = 12/4 + 2/4 + 1 = 18/4 \rightarrow b_{6,5}.
\end{aligned}
\qquad (8)
$$

Depois de 12 atualizações a matriz de *Betweenness* B alcançaria o estado descrito em 9.

$$D = \frac{1}{4}\begin{pmatrix} 0 & 18 & 2 & 0 & 2 & 18 \\ 18 & 0 & 18 & 2 & 0 & 2 \\ 2 & 18 & 0 & 18 & 2 & 0 \\ 0 & 2 & 18 & 0 & 18 & 2 \\ 2 & 0 & 2 & 18 & 0 & 18 \\ 18 & 2 & 0 & 2 & 18 & 0 \end{pmatrix} \quad (9)$$

Esse novo modelo matemático possibilita a análise da importância da conexão entre dois indivíduos com base na manutenção ou na construção de novas comunidades dentro de uma rede.

3.3. Visualização de Agrupamentos

Uma das maneiras mais interessantes e utilizadas para visualização das Redes Complexas é a representação de cores da matriz de vizinhança (ANDRADE et al, 2006). Para facilitar o entendimento, a representação de cores da matriz de vizinhança será referenciada como matriz de cores.

Com essa representação é possível identificar características de uma forma mais clara e imediata. Isso é de grande importância quando a rede observada contém um grande número de vértices e arestas. A matriz de vizinhança, $V_{i,j}$, é uma sobreposição de várias matrizes de adjacência que armazenam informação sobre distância, em número de arestas, entre dois vértices *i* e *j* (ANDRADE et al, 2006). A equação 10 define a matriz de vizinhança, onde *Mj* é a matriz de adjacência de ordem *l*, e *l* assumem valores de 1 até D (*l*=1,2,3,..., D).

$$MV = \sum_{l=1}^{D} lMj \qquad (10)$$

A figura 7 mostra a matriz de vizinhança e a matriz de cores do grafo ilustrado na figura 5. Observe as posições $V_{2,5}$, destacada em vermelho na figura 7, e $V_{5,1}$, destacada em verde na mesma figura. Elas definem a quantidade de arestas que existe no menor caminho entre os vértices 2 e 5 e entre os vértices 5 e 1 respectivamente.

A matriz de cores é uma forma alternativa de visualizar a matriz de vizinhança. Comparando as ilustrações da figura 7 percebe-se que as cores frias (cinza, azul escuro e azul claro) definem as regiões da matriz com um menor número de arestas e as cores quentes (verde, laranja e vermelho) identificam as regiões da matriz com um número maior de arestas. Esse é o raciocínio utilizado para identificar grupos ou *clusters* a partir da matriz de cores. As regiões mais conectadas, às que possuem uma quantidade maior de arestas, serão representadas pelas cores quentes e as menos conectadas, às que possuem uma quantidade menor de arestas, serão representados pelas cores frias. Isso possibilita a construção de uma estrutura modular e tornando a visualização mais natural (ANDRADE et al, 2009)

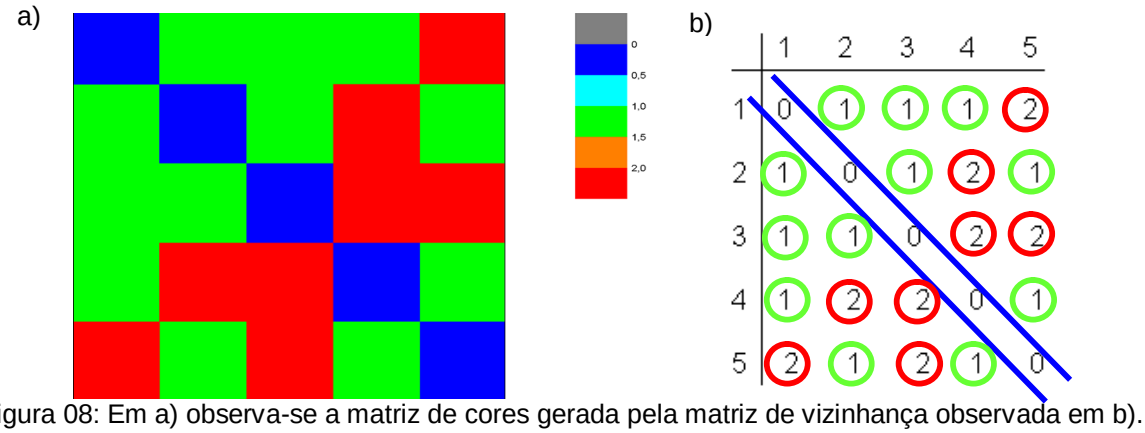


Figura 08: Em a) observa-se a matriz de cores gerada pela matriz de vizinhança observada em b).

Outra forma de identificar grupos em uma Rede Complexa é analisar os dendrogramas gerados pela análise de *Betweenness* (NEWMAN, GIRVAN; 2004). A sucessiva remoção de arestas de uma rede evidencia a formação de agrupamento de elementos. A figura 08 ilustra os resultados desse processo. As setas indicam o ponto onde um determinado número de arestas removidas provoca a formação de grupo. As barras têm o objetivo de facilitar a visualização do número de arestas removidas e os retângulos evidenciam os grupos formados. Uma possível leitura do dendrograma ilustrado na figura 08 seria a seguinte: após a remoção de 17 arestas o dendrograma evidenciou a formação de dois grandes grupos em vermelho. Esses grupos permanecem estáveis até a remoção da 33º aresta, que ocasionou a formação de dois grupos destacados em azul. Esse comportamento se repetiu após a remoção da 39º aresta, fato que deu origem a mais dois grupos destacados em verde. A rede permanece estável após a remoção da 59º aresta e, a partir desse ponto, não é possível observar a formação de novos grupos.

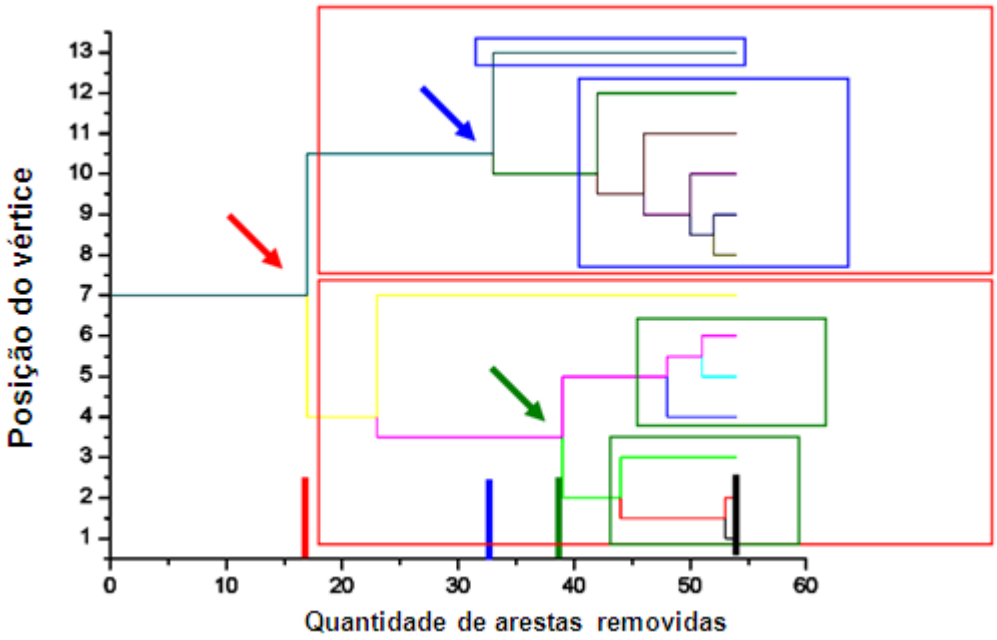


Figura 09: Dendograma. Destacam-se em vermelho, azul e verde os grupos realçados pela gradativa eliminação de arestas. As barras facilitam a visualização do número de arestas removido. Em preto, observa-se a barra que delimita o fim da formação de novos agrupamentos.

Conforme descrição expressa na sessão 3.1 é possível utilizar matrizes de adjacência para representar as redes complexas. Nesse caso os elementos *i* e j da matriz são 1 ou 0. Eles indicam a conexão, elemento 1, ou não, elemento 0, dos vértices da matriz. Por outro lado, o conceito de matriz de vizinhança diz que os elementos de uma matriz indicam a quantidade de arestas que existem no menor caminho existente entre dois elementos *i* e *j* (ANDRADE et al, 2006). O conceito de distância entre matrizes parte do princípio que dado duas matrizes de vizinhança distintas com o mesmo número de vértices, identificadas como ***α e β,*** é possível utilizar a Distância Euclidiana ***δ***(***α, β***) para definir a diferença entre as duas matrizes (ANDRADE et al, 2009). Essa diferença pode ser utiliza para identificar as arestas cuja eliminação causa a divisão da rede em comunidades (ANDRADE et al, 2009). Quanto menor o valor da Distância Euclidiana entre matrizes de vizinhança maior será a semelhança entre as comunidades que essas matrizes representam (BROWER; ZAR, 1977). A equação 11 ilustra a fórmula da Distância Euclidiana, onde p e q representam os elementos *i* e *j* da matriz de vizinhança.

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_x - q_x)^2} = \sqrt{\sum_{x=1}^{n} (p_x - q_x)^2} \qquad (11)$$

A figura 9 ilustra a comparação entre os resultados apresentados pela Distância Euclidiana ***δ*** e a função de modularidade Q proposta por (NEWMAN; GIRVAN, 2004) utilizando a rede social da academia de karatê Zachary. Nessa rede é possível perceber a formação de dois grupos sobre duas pessoas, o treinador e a secretária (Zachary, 1977). As duas técnicas, Distância Euclidiana ***δ*** e a função de modularidade Q, têm o objetivo detectar a mudança de comportamento em Redes Complexas

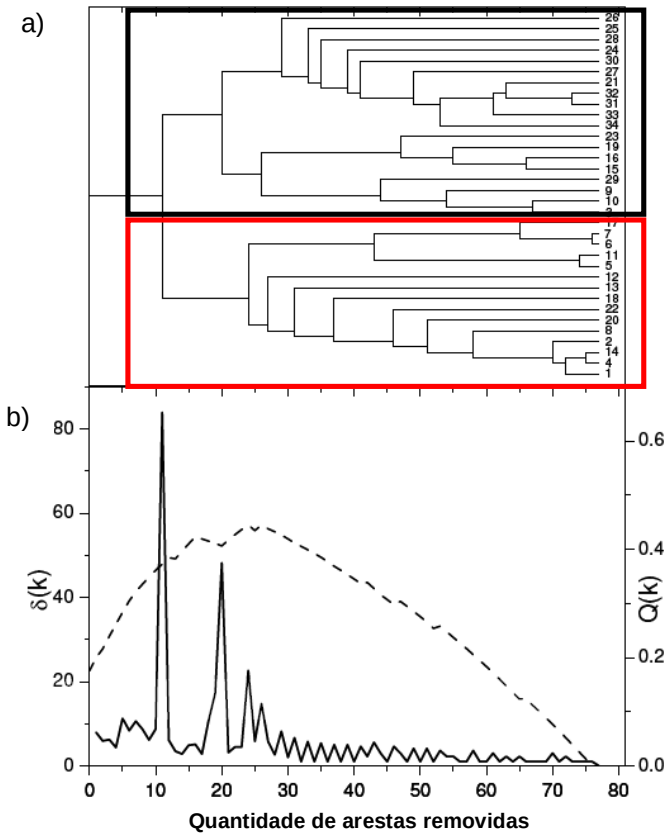


Figura 10: Em a) observa-se o dendograma construído pela eliminação consecutiva de arestas com maior grau de *betweenness*. Em b) observa-se os valores de δ (em linhas sólidas) e os valores de Q (em linhas pontilhadas) para as mesmas arestas eliminadas. Em a) destacam-se em preto e vermelho os dois grupos formados sobre o professor e a secretária do clube de Karatê Zachary (ANDRADE et al, 2009).

A Distância Euclidiana identifica esses pontos de formação de grupos, também chamados de limiar crítico com picos durante a sua leitura, fato que garante uma detecção eficiente da formação de grupos (ANDRADE et al, 2009). Observando os valores de δ(k) e Q(k) ilustrados na figura 9, é possível notar que existe uma relação entre os valores de δ e Q, mas eles não são equivalentes. Além disso, δ(k) mostrou-se mais sensível durante os eventos que concretizam a formação dos grupos (ANDRADE et al, 2009).

4. BIOINFORMÁTICA

A bioinformática é uma área multidisciplinar criada a partir da combinação da Química, Física, Biologia, Ciência da Computação e Matemática. Ela se desenvolveu devido à enorme massa de dados gerada pelos grandes projetos na área de genômica, muitos deles idealizados na década de 80 (POLANSKI; KIMMEL, 2007). Um dos grandes benefícios dos projetos de genoma e mais recentemente dos projetos de proteoma, é o estudo da estrutura molecular de proteínas e enzimas. A compreensão molecular da estrutura de alvos moleculares terá um papel cada vez mais representativo na definição de diagnósticos e tratamentos de doenças (SANTOS-FILHO, ALENCASTRO, 2003).

A bioinformática possibilita a aplicação de técnicas computacionais para armazenar, gerenciar e analisar dados biológicos. Essa última característica tornou a bioinformática uma grande aliada de pesquisadores de distintas áreas, possibilitando a construção de ferramentas que permitem a análise e a mineração de dados nas extensas bases de dados como UniProt, Swiss-Prot, TrEMBL, pIR, ProSite, Pfam e SMART. Essa parceria deu origem a novos métodos teóricos e continua respondendo perguntas que guiam os pesquisadores no desenvolvimento de tecnologia em prol do bem estar da humanidade (GIBAS; JAMBECK, 2001).

Atualmente, o grande problema que a bioinformática se dispõe a resolver é o reconhecimento de padrões e as técnicas mais utilizadas e aceitas pela comunidade científica são a análise de similaridade e a análise filogenética (GIBAS; JAMBECK, 2001).

4.1. Alinhamento de Sequências

Alinhamento de sequências é uma técnica utilizada para comparação de moléculas de várias espécies. Ela tem uma vasta área de aplicação que vai do estudo do grau de parentesco entre dois organismos até a prescrição minuciosa de um fármaco. Na grande maioria das vezes, as moléculas consideradas para

esse estudo são moléculas de DNA, RNA e proteínas. A natureza química desses compostos, polímeros, permite que elas sejam representadas por uma sequência de caracteres (figura 10), fato que possibilita que a comparação entre duas sequências seja concretizada pela análise entre os conjuntos de caracteres que as representam (BRITO, 2003). A figura 11 ilustra o alinhamento entre duas sequências protéicas, AAB24881 e AAB24882. Os asteriscos (*) representam os pontos de identidade entre as sequências, os dois pontos (:) representam as diferenças entre as sequências.

```
AAB24882        PSHLQYHERTHTGEKPYECHQCGQAFKKCSLLQRHKRTHTGEKPYE-CNQCGKAFAQ
AAB24881        HSHLQCHKRTHTGEKPYECNQCGKAFSQHGLLQRHKRTHTGEKPYMNVINMVKPLHN
                 ****:*:***********:***:**::::*****************:::::::*::::
```

Figura 11: Exemplo de alinhamento entre duas sequências produzido pelo programa ClustalW (THOMPSON, 1994)

Encontrar o alinhamento entre sequências e, com isso, determinar a similaridade entre elas, é um dos problemas mais fundamentais da Bioinformática. Os dados gerados nessa comparação podem ser utilizados para construir uma árvore filogenética ou até determinar a estrutura secundária e terciária de uma proteína, possibilitando o estudo da sua função (BRITO, 2003).

4.1.1. Alinhamento de pares de sequências

Inicialmente será apresentado o alinhamento entre pares de sequências. Para isso, considere as seguintes sequências ATGGCCTC e ATGGCGC. Embora o alinhamento mostrado na figura 12 não ressalte, uma breve inspeção nas sequências evidencia a semelhança que existe entre elas. Esse problema pode ser resolvido com um alinhamento melhor.

```
ATGGCCTC_____
_______ATGGCGC
```

Figura 12: Um alinhamento entre ATGGCCTC e ATGGCGC (BRITO, 2003)

Observado o segundo alinhamento ilustrado na figura 13, fica claro que as sequências têm muito em comum. As diferenças entre elas é uma base G na primeira sequência e a ausência da base T na segunda sequência. Essas observações nas sequências acima estudadas têm um significado biológico. As inserções, remoções e substituições de nucleotídeos são resultados de mutações oriundas do processo de evolução. Com isso é possível introduzir o conceito de "qualidade" ou "pontuação" de um alinhamento que é a ponderação feita entre as combinações de nucleotídeos com o objetivo de expor de uma maneira mais clara as semelhanças entre as sequências. A figura 13 ilustra um alinhamento "melhor" que o alinhamento da figura 12 (BRITO, 2003).

```
ATGGCCTC
ATGGCG_C
```

Figura 13: Um alinhamento "melhor" entre ATGGCCTC e ATGGCGC (BRITO, 2003)

A pontuação de um alinhamento é determinada por uma função que associa a cada par de símbolos formados entre as sequências uma pontuação. Essa função é chamada de Função de Pontuação. Como na grande maioria das vezes as funções de pontuação são representadas por matrizes, elas também são conhecidas como Matrizes de Pontuação. Esse procedimento funciona da seguinte maneira: o alinhamento A entre duas sequências s e t sobre um alfabeto Σ é determinado pela função p: Σ' x Σ' → Q, e para cada símbolo (*i*, *j*) dessa matriz, é associada uma pontuação (BRITO, 2003). De uma forma mais didática temos: o alinhamento A[*i*, *j*] entre duas sequências ATGGCCTC e ATGGCGC sobre um alfabeto (A, G, T, C) é determinado pela função p: f(*x*) = y, e cada símbolo (*i*, *j*) dessa matriz recebe o valor *y* da função p que representa o valor da pontuação. Os valores que y pode assumir dependem da matriz e do modelo do alinhamento. A equação 12 define formalmente o alinhamento entre as sequências *s* e *t*:

$$(A) = \sum p(s'[j], t'[j]) \qquad (12)$$

O alinhamento A é igual ao somatório dos pesos entre cada par da matriz A[*i*, *j*], tendo *s'* e *t'*, respectivamente, as sequências *s* e *t* com as possíveis inserções, remoções e substituições (figura 14).

Existem três modelos que podem ser utilizados em alinhamentos de sequências: o modelo Local, o modelo Global e o modelo Semi-Global. Eles têm o mesmo objetivo, mas são aplicados em situações diferentes. O alinhamento Local é útil quando as duas sequências têm tamanhos próximos. Por outro lado o alinhamento Global é bem aplicado em sequências de tamanhos diferentes e que possuem poucas evolucionariamente bem sucedidas, também chamadas de regiões conservadas. O enfoque Semi-Global, também conhecido de pontas livres, é similar ao global. Eles são utilizados para encontrar fragmentos idênticos desprezando os espaços nos extremos da sequência (CARAZZOLLE, 2008).

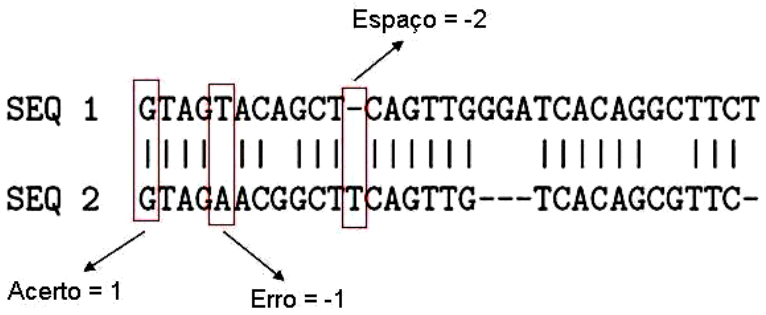


Figura 14: Exemplo de um alinhamento entre duas sequências. *Score* = Acertos (1) + Erros (-1) + Espaços (-2) = 24 – 4 – 10 = 10 (CARAZZOLLE, 2008).

4.1.2. Alinhamento de várias sequências

O problema proposto pelo alinhamento de várias sequências é uma generalização do problema de alinhamento entre pares de sequências. Esse conceito possibilita estudar um dos problemas mais importantes da Biologia Computacional: a construção das Árvores Filogenéticas (figura 15). Com elas, podemos estudar as relações de descendência e ascendência entre espécies através de diagramas (BRITO, 2003).

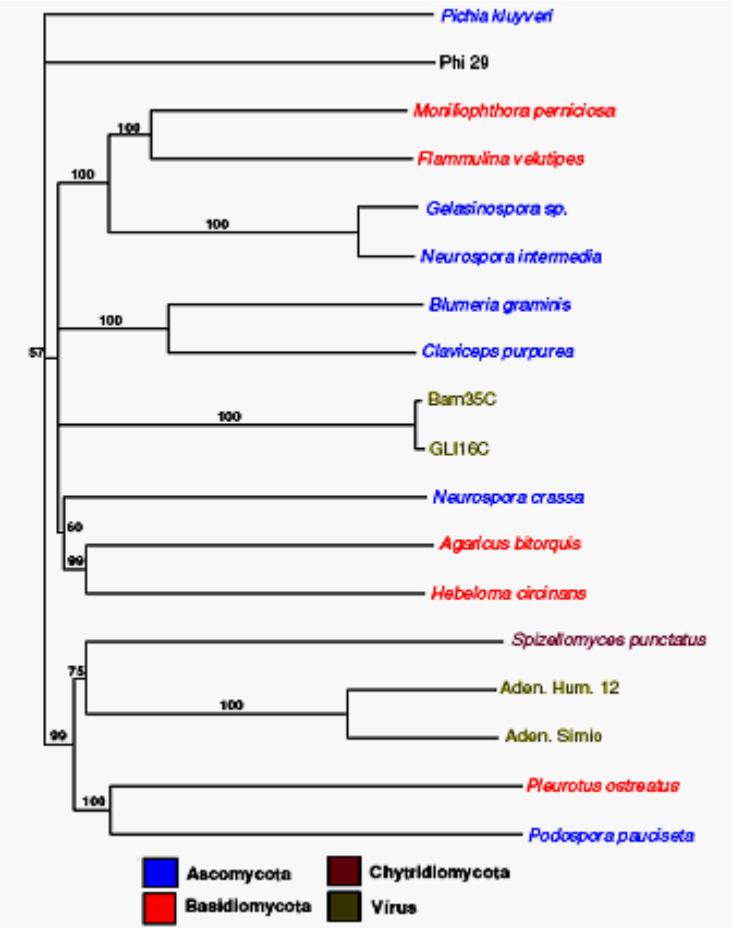


Figura 15: Árvore Filogenética gerada pela análise de Distância, utilizando sequências aminoacídicas de plasmídeos fúngicos e virais. Números acima dos ramos correspondem aos valores percentuais de *bootstrap* (ANDRADE, 2008).

A etapa inicial de qualquer um dos métodos de análise filogenética é o alinhamento entre as sequências que serão estudadas. Posteriormente, as árvores filogenéticas são construídas. Nelas, cada sequência é representada por uma folha e os ramos das árvores conotam as relações de parentesco entre elas. Neste estudo, os resultados mais interessantes são também os mais inesperados como, por exemplo, a proximidade entre fungos e vírus (ANDRADE, 2008) ou o parentesco entre espécies geograficamente separadas e morfologicamente diferentes geralmente abordadas no estudo de genética de populações (HARTL, 1999).

O objetivo do alinhamento de várias sequências (figura 16) e do alinhamento de pares de sequência é o mesmo: obter bons alinhamentos. Para isso uma pontuação, ou custo, é atribuído a cada alinhamento e dessa forma é

possível escolher os melhores alinhamentos. Seguindo a mesma abstração do alinhamento de pares de sequências, o alinhamento de várias sequências é representado por uma matriz e é definido como: dado um número inteiro k >= 2 e k sequências *s1*,..., *sk* formadas por um alfabeto Σ não-nulo, um alinhamento A de *s1*,..., *sk* é uma matriz A = A[*i*, *j*] de dimensões *k* x *k* com entradas Σ' = Σ U { - }, onde { - } representa as possíveis inserções e/ou deleções nas sequências.

```
Oryctalagus cuniculus (K03256)      -AAGGTGAATG GGA GAA
Homo sapiens (U01317)               CAAGGTGAACGTGGATGAA
Capra hircus (M15387)               -AAGGTGAAAGTGGA-GAA
Mus musculus (J00413)               AAAGGTGAA-TCCGATGAA
Gallus gallus β-A (L17432)          -AAGGTCAATGTGGCCGAA
Gallus gallus β-H (L17432)          CAAGGTCAATG-GGCCGAA
                                     ***** **      *   ***
```

Figura 16: Alinhamento de várias sequências construído pelo programa Clustal W. (BRITO, 2003)

O método utilizado para determinar a pontuação define a qualidade do alinhamento entre várias sequências e é semelhante ao utilizado no alinhamento de pares. Uma pontuação é atribuída a cada coluna da matriz, figura 17. A pontuação do alinhamento é a soma das pontuações das colunas, figura 18. A pontuação da coluna é determinada pela função de Soma de Pares (SP): ela mapeia uma coluna C com uma quantidade *k* de caracteres a sua pontuação. A equação 13 define a função SP onde C[*i*] representa os caracteres da coluna C (BRITO, 2003).

$$SP(C) = \sum_{1 \le i \le i' \le k} c(C[i], C[i']) \qquad (13)$$

Figura 17: Exemplo de um alinhamento entre várias sequências. Cada coluna tem uma pontuação. Pontuação da coluna vermelha = 10, pontuação da coluna verde = 15, pontuação da coluna azul =20, pontuação da coluna amarela =25 e pontuação da coluna preta = 30.

A pontuação SP(A) de um alinhamento A das sequências *s1*,..., *sk* é definida pela equação 14, onde A[*i*, *j*] representam as colunas de A e *l* é o número de linhas de A.

$$SP(A) = \sum_{j=1}^{l} \sum_{1 \le i < i' \le k} c(A[i, j], A[i', j]) \qquad (14)$$

Figura 18: Exemplo de um alinhamento entre várias sequências. O valor das colunas é sumarizado gerando a pontuação do alinhamento. *Score* = Pontuação da coluna vermelha + pontuação da coluna verde + pontuação da coluna azul + pontuação da coluna amarela + pontuação da coluna preta. *Score* = 10 +15 +20 +25 +30 = 100.

4.2. Análise de similaridade

A partir da Análise de Similaridade é possível determinar o grau de identidade entre sequências, expresso em percentual (%). A identidade entre sequencias e expressa pela fórmula 15, onde *a* representa o número de pareamentos e *b* representa tamanho da sequência (GIBAS; JAMBECK, 2001).

$$I = \frac{a}{b} \qquad (15)$$

Analisando o percentual de identidade entre as sequencias, é possível definir se existe um ancestral comum entre elas, ou seja, se as sequencias são homólogas. O conceito de Homologia refere-se a um modelo biológico de evolução, é uma hipótese evolucionária baseada no grau de identidade entre duas ou mais sequências. Defini-se como identidade o processo de ajustes das sequências, buscando o alinhamento entre elas, com o objetivo de maximizar as suas semelhanças (GIBAS; JAMBECK, 2001).

Na sessão anterior 4.1 foi descrito um conjunto de regras que gerenciam a "edição" de sequências, visando aproximar esses objetos de estudo dando um valor ao final do processo. Em outras palavras, esse processo penaliza ou gratifica os alinhamentos com base em um sistema de escores que representa um modelo evolutivo, graduando em pontos as possíveis mutações (inserções ou deleções) aminoacídicas. Essas ferramentas matemáticas são chamadas de Matrizes de Distância ou Matrizes de *Score*. Como cada matriz representa uma teoria evolutiva diferente, sua escolha tem uma forte influência no resultado da análise. Uma série de critérios é levada em consideração durante a definição dessas matrizes, ente eles destacam-se as propriedades químicas associadas às cadeias aminoacídicas ou protéicas (GRANTHAM, 1974; RAO, 1987), as frequências de substituição observadas a partir de proteínas evolutivamente próximas (DAYHOFF et al 1978; MCLACHLAN, 1971) e a frequência de aparição de cada um dos aminoácidos em uma estrutura secundária (LEVIN et al 1986; RAO 1987). Os dois conjuntos de modelos mais conhecidos e utilizados de Matrizes de *Score* são o PAM e o BLOSUM.

A matriz PAM (*Point Accepted Mutation*) foi a primeira a ser construída. O seu funcionamento se baseia nas seguintes hipóteses: os eventos mutacionais são independentes do contexto, um acontecimento mutacional numa certa posição é independente dos eventos mutacionais anteriores e a probabilidade de substituição de *X* em *Y* é a mesma que a de *Y* em *X* (SCHWARTZ, DAYHOFF, 1978). A PAM foi construída com base em 1572 sequências protéicas divididas em 71 famílias. Para evitar problemas de substituição foram utilizadas sequências com um grande número de regiões conservadas, com mais de 85% de

similaridade entre elas. Para aumentar a representatividade da matriz e poder simular um número maior de mutações é possível multiplicar a matriz PAM1 por ela mesma dando origem a matriz PAM2. Essa operação pode ser realizada sucessiva vezes com o objetivo de estimar de uma maneira mais precisa a distância evolutiva entre as sequências. Geralmente utiliza-se a matriz PAM250, porém, quando as sequências estudadas representam espécies que divergiram recentemente a matriz PAM125 é mais utilizada (ROCHA, 2007).

Por outro lado, existem características criticáveis na matriz PAM. O modelo evolutivo é muito simplificado, as substituições dos aminoácidos não são independentes e equidistantes e os erros existentes nas matrizes derivadas se propagam em uma escala exponencial. Caso exista um erro na matriz PAM1 ele será potencialmente grande na matriz PAM125. Para resolver esses problemas utiliza-se atualmente uma matriz que tem como base a probabilidade de substituições diretamente das próprias sequências, a matriz BLOSUM (*BLocks SUbstitution Matrix*). A primeira delas foi a BLOSUM62, ela representa as substituições entre sequências aminoacídicas com identidade menor que 62%. De forma semelhante existe a BLOSUM50 e a BLOSUM80. A eficiência desse novo modelo ganhou credibilidade e foi bem aceita pela comunidade acadêmica, e devido a isso, várias ferramentas foram desenvolvidas implementando essas matrizes para concretizar suas análises. Um bom exemplo é o BLAST, *Basic Local Alignment Search Tool*, que utiliza, por padrão, a matriz BLOSUM62 (ROCHA, 2007).

A figura 19 faz uma comparação entre o comportamento e aplicabilidade das diferentes matrizes de *score*. Nela é possível notar que à medida que o número de multiplicações entre as matrizes PAM aumenta mais divergente, ou seja, menos semelhante, as matrizes PAM serão das matrizes BLOSUM que representam substituições entre sequências aminoacídicas com menor identidade. Isso é justificável devido à propagação de erros ocasionado pelas sucessivas multiplicações entre as matrizes PAM (WARD, 2009).

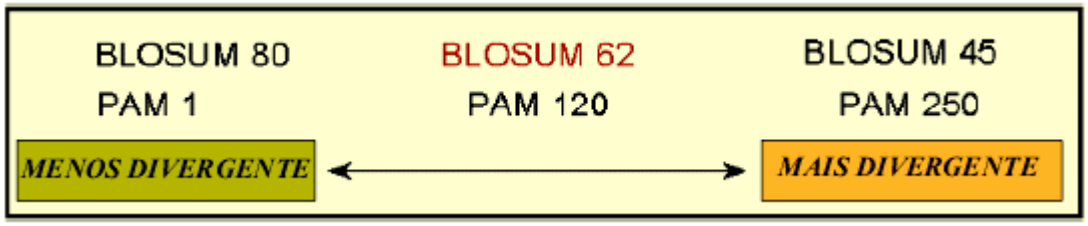


Figura 19: Comparação entre as matrizes *score* PAM, BLOSUM e suas relações de divergência. Em vermelho destaca-se a matriz utilizada pelo BLAST (WARD, 2009).

4.3. Análise Filogenética

A análise filogenética é a mais poderosa ferramenta de análise estatística dentre os modelos disponíveis atualmente para ciência (MAR et al., 2005). Construir relações filogenéticas significa definir relações hierárquicas entre indivíduos a partir de sequências de DNA ou aminoácidos. Para isso, dois passos importantes devem ser executados: o alinhamento das sequências, que tem o objetivo de encontrar homologias ou diferenças entre as sequências e a criação de um modelo matemático que descreva a evolução das sequências. Os modelos podem ser criados empiricamente ou utilizando propriedades químicas ou biológicas das moléculas de DNA e de aminoácidos. Esses modelos possibilitam determinar a distância genética entre duas sequências a partir do número de substituições de nucleotídeos encontrados em determinadas regiões ou sítio das sequências. Essas substituições são derivadas dos diversos cruzamentos que formaram as linhagens dos ancestrais comuns e atuais das espécies analisadas.

Para aumentar a representatividade dos dados e facilitar a análise filogenética, os dados gerados são dispostos em uma árvore, a Árvore Filogenética (figura 15). Nesta árvore, as sequências homólogas ficam nas pontas dos ramos e os nós internos da árvore representam o grau de ancestralidade entre as sequências homologas. Utilizar um método estatístico bem definido é um passo muito importante para encontrar a topologia da árvore que melhor descrevem as relações filogenéticas entre as sequências. As técnicas mais utilizadas na análise filogenética são *Neighbor-joining* (NJ) ou Distância, Parcimônia, Máxima Verossimilhança (*Maximum-likelihood* – ML) e a estatística bayesiana (ANDRADE, 2008).

O algoritmo de NJ é o mais comum e popular entre as técnicas, uma vez que é simples e veloz, entretanto, ele apresenta uma boa *performance* quando o grupo de sequências apresenta pequena divergência entre si. Ele é rápido e utiliza uma matriz para representar a distância genética entre as sequências de DNA ou proteína. A fraqueza apresentada pelo Neighbor-joining é que as diferenças apresentadas entre as sequências não refletem de maneira satisfatória a distância evolutiva entre elas. Devido a isso o NJ é bem aplicado em árvores filogenéticas que apresentam sequências que divergiram recentemente. Esse algoritmo não é recomendado para definir relações de parentesco antigas (ANDRADE, 2008).

A análise de parcimônia parte do princípio que nucleotídeos ou aminoácidos que sofreram mutações há pouco tempo, também chamados de caracteres apomórficos, compartilhadas por diferentes grupos provêm de um ancestral comum e exclusivo. Dessa forma, grupos de indivíduos são formados com base em um ou mais caracteres apomórficos compartilhados entre os indivíduos de diferentes filos. Essa é a base conceitual na análise de parcimônia (GOLDANI; CARVALHO, 2003).

Com a possibilidade de existir uma série de caracteres apomórficos, diferentes agrupamentos podem ser igualmente plausíveis perante a análise de parcimônia e, consequentemente, diversas árvores filogenéticas seriam igualmente possíveis. Para resolver esse problema, existe o conceito de árvore de consenso estrito. Essa árvore especial contém topologias que não contradizem as árvores iniciais. Caso não exista ancestralidade comum exclusiva, não existe congruência entre as árvores iniciais, dessa forma os dados utilizados para a construção da árvore não possuem informação filogeneticamente válida (HARRISON, LANGDALE, 2006).

Aplicações baseadas em verossimilhança são poderosas ferramentas para inferir árvores filogenéticas, mas são computacionalmente intensas. Devido a isso, uma análise de máxima verossimilhança pode ser lenta para problemas que envolvem um grande número de sequências alinhadas e/ou muitas replicações no *bootstrap* (reamostragem com reposição), parâmetro que permite verificar o suporte estatístico entre os agrupamentos de sequências e a relação entre os

ramos das árvores (MAR et al., 2005). Apesar disso, este método de inferência reconstrói muito bem relações entre sequências separadas por um longo período de tempo, ou que evoluíram muito rapidamente, além de ser o método estatisticamente mais robusto. Na máxima verossimilhança, a árvore filogenética é gerada a partir do melhor resultado (maior probabilidade) entre as sequências observadas. Para utilizar a análise de máxima verossimilhança é necessário utilizar um modelo evolutivo específico para o grupo de indivíduos estudado. Este modelo descreve a probabilidade de eventos que ocorrem em sequências dispostas nas extremidades das árvores (HOLDER; LEWIS, 2003).

A análise bayesiana é a mais nova abordagem dentre os métodos de análise filogenética. Porém o que mais chama a atenção nesse método estatístico são os produtos gerados na sua análise. Muito relacionado ao método de Máxima Verossimilhança, a análise Bayesiana além de inferir uma árvore filogenética, gera um grau de incerteza relacionado a cada grupo da árvore, uma vez que trabalha com probabilidades condicionais (HOLDER; LEWIS, 2003).

A análise filogenética bayesiana não implementa apenas o algoritmo padrão de MCMC, *Markov Chain Monte Carlo*, que simula a distribuição hierárquica entre elementos baseado em um modelo previamente conhecido, mas também sua variante chamada de *Metropolis-coupled Markov Chain Monte Carlo* ou MC^3. Essa variante proporciona um melhor aproveitamento das sucessivas árvores que são geradas durante a análise bayesiana. Isso é possível por que ao fim de cada cadeira, ou rodada, duas árvores são trocadas de posição dentro da estrutura hierárquica que esta sendo construída, a Árvore Filogenética. Essa troca é aceita se a nova estrutura gerada a partir da troca atingir um valor de aceitação informado pelo usuário, caso contrário a troca não é realizada. Esse procedimento se repete até que o número total de cadeias informado pelo usuário seja atingido. (HUELSENBECK; RONQUIST, 2001).

4.4. Domínios Conservados

Domínios são unidades tridimensionais de proteínas que apresentam uma função e/ou estrutura específica(s). Eles são identificados através de padrões em sua sequência (através de alinhamento de sequências) e em sua estrutura secundária e terciária (através de comparação de estruturas tridimensionais) (MARCHLER-BAUER et al., 2009).

Domínios conservados contêm padrões específicos de sequências que foram conservados ao longo do processo evolutivo e, portanto, representam os blocos de construção na evolução molecular de proteínas e que são recombinados em diferentes arranjos para formar proteínas com diferentes funções (MARCHLER-BAUER et al., 2009).

A identificação e caracterização de domínios conservados em proteínas é uma ferramenta poderosa de análise que auxilia bastante o entendimento das relações entre sequência/estrutura/função em proteínas relacionadas (MARCHLER-BAUER et al., 2009), permitindo o desenho de diferentes inibidores para alvos moleculares em enzimas específicas.

5. MATERIAIS E MÉTODOS

Os passos sequenciais utilizados para a caracterização e análise das sequências protéicas da sintase da quitina através de uma abordagem comparativa, filogenética e de redes complexas são sintetizados no fluxograma mostrado na Figura 20.

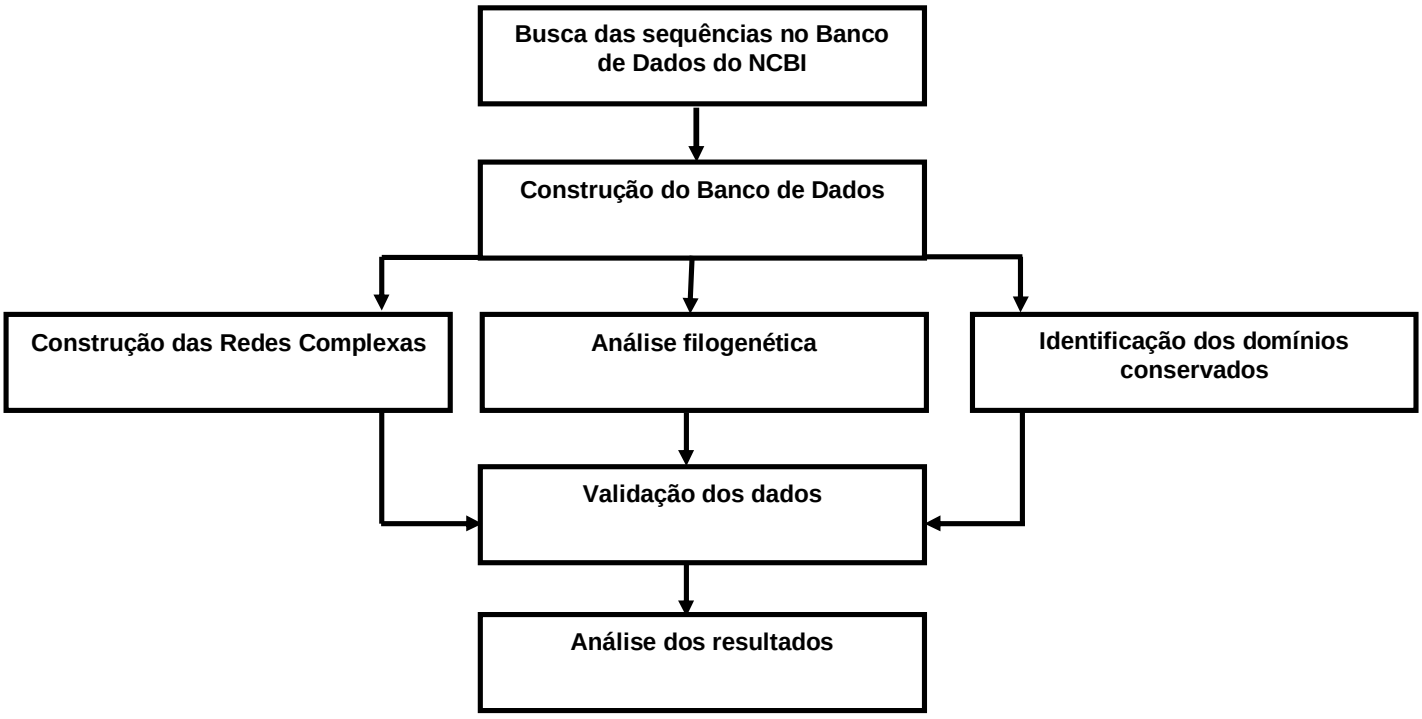


Figura 20: Fluxograma da metodologia utilizada para a análise das sequências protéicas da sintase de quitina.

5.1. A linguagem PERL

Existem diversas linguagens de programação que podem ser utilizadas para desenvolver aplicações na área de Bioinformática. Dentre elas, a linguagem Perl, *Pratical Extraction and Reporting Language*, uma linguagem estável, eficiente e multiplataforma que oferece ao programador extensões com outras linguagens de programação e interfaces com bancos de dados robustos e largamente

utilizadas em ambientes acadêmicos e comerciais. Tomando-se o ponto de vista do programador, é necessário menos tempo de programação para analisar dados utilizando a PERL do que linguagens como C ou Java (GIBAS; JAMBECK, 2001). Dados biológicos são armazenados em Bancos de Dados ou em arquivos texto. A linguagem PERL tem a capacidade de trabalhar com reconhecimento de padrões. Ela é munida de um grande conjunto de funções para tratamento de caracteres e possui bibliotecas que facilitam a construção de expressões regulares (DEITEL et al, 2002; GIBAS; JAMBECK, 2001). Além disso, a linguagem PERL possui uma sintaxe simples, flexível e de entendimento direto. Isso facilita o aprendizado tanto para que não tem experiência com desenvolvimento de sistemas como para pessoas que já trabalharam com outras linguagens de programação. Além disso, PERL é uma linguagem portável, ela foi concebida para funcionar em várias plataformas e ser integrada a bibliotecas específicas para Bioinformática como, por exemplo, a biblioteca BioPerl. Essas características tornam a linguagem PERL ideal para desenvolver aplicação para Bioinformática (GIBAS; JAMBECK, 2001).

5.2. Banco de Dados Relacional

Banco de Dados Relacional é um sistema de manutenção de registros que organiza as suas informações formalmente em tabelas que podem ser acessadas e modificadas de diferentes maneiras com o objetivo de reorganizar ou visualizar os dados que elas armazenam (CODD, 1970). Esse conceito foi criado para facilitar compartilhamento de grandes Bases de Dados de uma forma mais simples, garantindo o entendimento mais direto da representação interna dos dados.

Nesta nova abordagem, as relações (tabelas) são entidades que armazenam dados. Elas são compostas por atributos (colunas) e cada tupla (linha) que as relações contêm são os registros que elas armazenam. Dessa forma o total de dados que estão armazenados em uma Base de Dados deve ser encarada como uma coleção que varia em relação ao tempo, dependendo das operações de

inserção (*insert*) de novos registros e deleção (*delete*) ou alteração (*update*) de registros existentes (CODD, 1970).

Entre os atributos de uma relação, um (ou um conjunto deles) assumem valores únicos e são utilizados para identificação da tupla. Esse atributo é chamado de Chave Primária. A necessidade de criar referências cruzadas entre atributos de relações diferentes determinou a criação das Chaves Estrangeiras. Ela é a concretização do vínculo entre duas ou mais tabelas. Considere duas relações A e B, essas relações têm chaves primárias a' (da relação A) e b' (da relação B) respectivamente. Partindo do preceito que os a' e b' são atributos da relação B, podemos dizer que a' é uma chave estrangeira. Observe que a' pode (ou não) fazer parte da chave primária de B, mas necessariamente b' é chave primária de B (CODD, 1970) (figura 21).

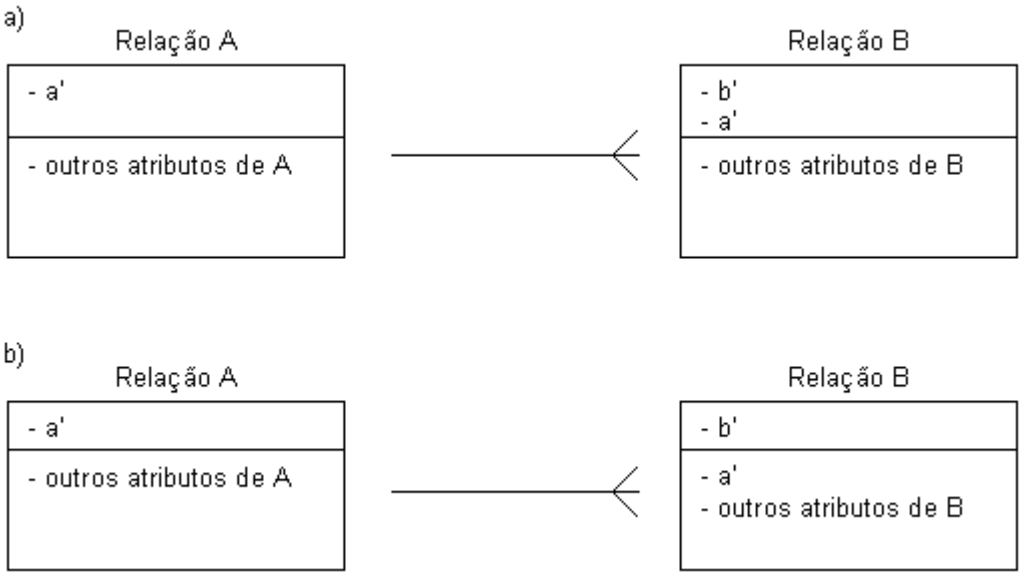


Figura 21: Referência cruzada entre as relações A e B. Em a) observa-se a presença de a' compondo a chave primária da relação B. Em b) o atributo a' não faz parte da chave primária de B. Em ambas as situações o atributo a' é uma chave estrangeira.

Sendo uma subárea da Ciência da Computação, os Sistemas de Gerenciamento de Banco de Dados, SGBD, (figura 22) tem como objetivo o estudo dos problemas relacionados à gerência de grandes bases de dados. O termo "grande" é informal e está diretamente relacionado ao poder de armazenamento disponível (SILBERSCHATZ; HENRY, 1996). O conceito de

Banco de Dados é bem aceito pela comunidade acadêmica e pelo mercado de sistemas. Isso é observado pela quantidade significativa de produção teórica nessa área e pelo impacto comercial que essas ferramentas têm no mercado de software (SILBERSCHATZ; HENRY, 1996).

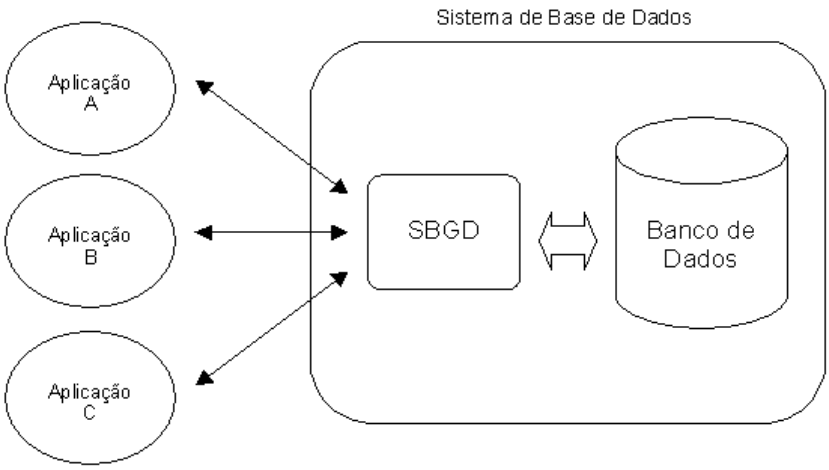


Figura 22: Sistema de Gerenciamento de Banco de Dados (SGBD).

Os dados que estão armazenadas em Bancos de Dados Relacionais são manipulados pela linguagem *Structured Query Language* (SQL). Ela foi desenvolvida na década de 70 por Codd e tem como base estrutural os conceitos da álgebra relacional. Desde a sua criação, diversos “dialetos” da linguagem SQL foram criados. Isso ocorreu devido às modificações que os fabricantes de Banco de Dados implementavam para facilitar o desenvolvimento de aplicações destacando o seu produto no mercado. Devido a isso, a necessidade de criar um padrão para essa linguagem se tornou iminente, e em 1986, a *American National Standards Institute* (ANSI) criou o SQL ANSI (SILBERSCHATZ, 2006). Esse padrão foi revisado três vezes (1992, 1999, 2003) com o objetivo de acompanhar a evolução dos Bancos de Dados e implementar os novos conceitos da Engenharia de *Software*. As versões mais atuais da linguagem SQL já implementam objetos de Banco de Dados que prezam pelo desempenho do Banco (*procedures*), segurança (*views*) e consistência de dados (*triggers*) (SILBERSCHATZ, 2006).

5.2.1. Construção de Banco de Dados

O conjunto de registros que estão armazenados no Banco de Dados é composto por sequências protéicas completas e parciais, que sintetizam a enzima sintase da quitina em fungos basidiomicetos. Elas foram obtidas da base de dados do NCBI (http://www.ncbi.nlm.nih.gov/) através de buscas textuais, utilizando palavras-chave com os operadores booleanos conforme ilustrado na figura 23. As sequência obtidas refletem o estado do banco de dados até o dia 11/09/2009.

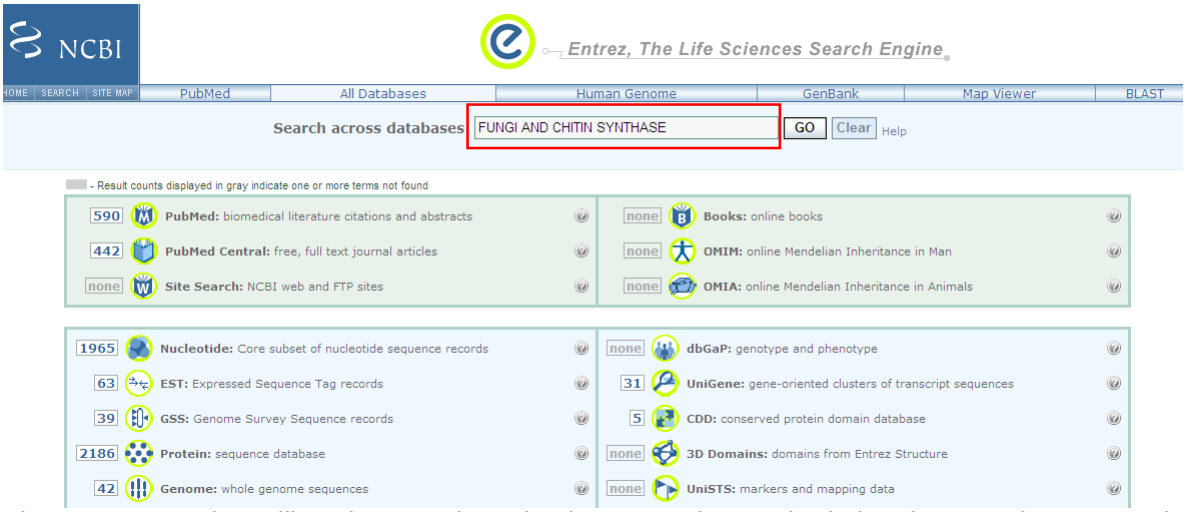


Figura 23: Pesquisa utilizando operadores booleanos no banco de dados do NCBI (NCBI, 2007). Em vermelho destacam-se as palavras-chave e operadores utilizados em uma busca.

Para obter mais informações sobre as sequências e aumentar o caráter qualitativo dos registros do Banco de Dados, foi utilizado o formato de arquivo 'GenBank'. Este tipo de arquivo disponibilizado pelo NCBI possui um conjunto de atributos muito rico (classificação da enzima, classificação taxonômica, palavras-chave entre outros), fato que possibilita a análise *ad-hoc* das sequências caso haja necessidade. A tabela 01 ilustra a classificação dos registros mais importantes do banco de dados. Defini-se como sequencias completas as sequencias oriundas do sequenciamento completo de uma amostra de tecido. As sequencias parciais são fragmentos de sequencias, elas são pedaços do sequenciamento de uma amostra de tecido (GIBAS; JAMBECK, 2001). Mais detalhes sobre os registros do banco de dados encontram-se no apêndice J.

Tabela 01 – Estatística do Banco de Dados

Tipo	Quantidade
Total de sequências	230
Nº de sequências completas	39
Nº de sequências parciais	191
Nº total de espécies	54

Os arquivos que continham as sequências protéicas foram processados por scripts escritos utilizando a linguagem de programação PERL. Para auxiliar o desenvolvimento dos scripts PERL foi utilizado uma biblioteca chamada Bioperl (LETONDAL, 2007). Ela contém um conjunto de funções escritas em PERL que facilitam a manipulação dos registros obtidos nos arquivos do NCBI. O SGBD utilizado nesta pesquisa foi o MySQL 5.0. Esta ferramenta de gerenciamento de arquivos implementa todas as características que um SGBD robusto necessita para garantir a persistência e a segurança dos dados.

A base de dados desta pesquisa é composta por duas tabelas (figura 24). Uma tabela chamada SEQUENCIA armazena as sequências que foram alvo de estudo. A tabela SIMILARIDADE armazena o grau de similaridade entre as sequências. Para facilitar a manipulação do grande número de registros que os scripts PERL iriam manipular foram criadas duas *procedures*, um para efetuar inserções na tabela SEQUENCIA e outra para efetuar inserções na tabela SIMILARIDADE. Logo, o código fonte dos scripts tornou-se mais legível, aumentando a capacidade de manutenção das suas linhas código.

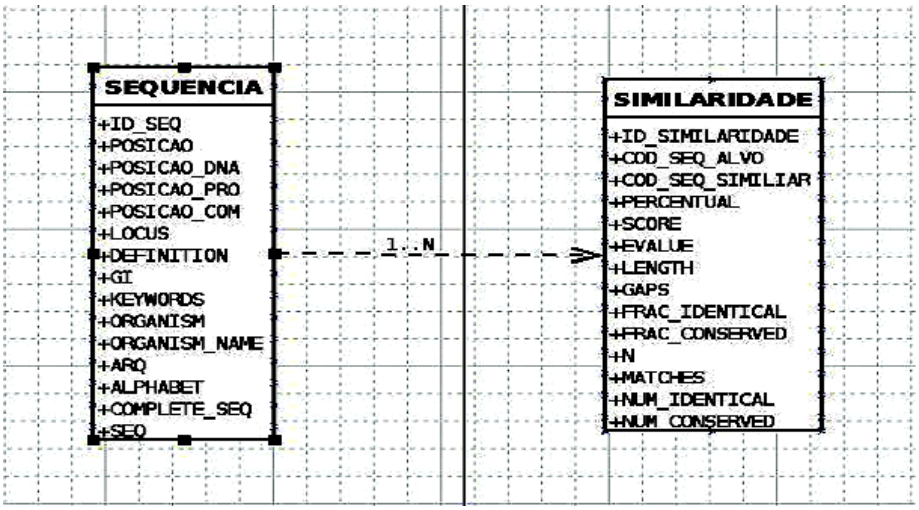


Figura 24: Estrutura do Banco de Dados (Tabelas Sequência e Similaridade)

A utilização da biblioteca Bioperl foi guiada pela API (*Application Programming Interface*) disponível em (http://www.bioperl.org/wiki/API_Docs). As instruções SQL utilizadas para criação, atualização e manipulação das informações do banco de dados estavam no padrão ANSI (*American National Standards Institute*). O código fonte dos scripts SQL utilizado na criação do banco de dados encontra-se no apêndice A.

5.3. Construção das Redes Complexas

A construção das Redes Complexas compreendeu três etapas: (i) análise de similaridade entre as sequências através da ferramenta BLAST, *Basic Local Alignment Search Tool* (ALTSCHUL et al., 1997), (essa análise foi subdividida por tipo de sequências), (ii) construção da matriz de similaridade, (iii) construção das matrizes de vizinhança (as matrizes foram criadas por grau de similaridade).

Para efetuar a análise de similaridade foi utilizado a ferramenta *StandAlone* BLAST. Essa versão da ferramenta BLAST possibilita executar buscas BLAST em bases de dados locais. Existem outras opções como as versões Network BLAST e BLAST URL API que possibilitam a execução das buscas BLAST em modo on-line através de uma interface de programação ou utilizando o protocolo HTTP, *Hypertext Transfer Protocol*, mas a quantidade de operações que seriam executadas, a latência que existiria entre elas e os requisitos funcionais, como o

tempo de conexão no servidor entre outros, faria com que essas versões de execução on-line fossem muito custosas computacionalmente, fato que torna a versão *StandAlone* BLAST (ALTSCHUL et al., 1997) a melhor opção. Buscando aumentar a qualidade dos dados gerados pela análise de similaridade, foi utilizado um Valor E (*E-value*) variando entre 10^{-5} e zero, que são considerados estatisticamente significativos. Essas ferramentas podem ser obtidas no endereço (ftp://ftp.ncbi.nih.gov/blast/executables) e estão disponíveis para vários tipos de arquitetura dentre as quais destacam-se a LINUX, Macintosh, Win32(PC) e Solaris.

Seguindo as instruções do manual da ferramenta *StandAlone* BLAST, o conjunto de dados armazenado na tabela SEQUENCIA foi replicado para constituir a base de dados da ferramenta BLAST. Cada sequência que estava armazenada na tabela SEQUENCIA foi comparada com as sequências da base de dados da ferramenta BLAST e os resultados dessas iterações foram armazenados na tabela SIMILARIDADE. Essas iterações geraram um grande número de registros, fato que dificultaria as operações necessárias para gerar as Redes Complexas a partir dos dados armazenados na tabela SIMILARIDADE. O código fonte dos scripts PERL utilizados para inserir dados nas tabelas SEQUENCIA e SIMILARIDADE encontram-se nos apêndices B e C respectivamente. Para poupar tempo computacional, as informações que representavam o grau de similaridade entre as sequências foram dispostas em uma matriz, a matriz de similaridade S (figura 25), onde a posição S[i, j] armazena o grau de similaridade entre as sequências *i* e *j*. A figura 26 ilustra um exemplo de uma matriz de similaridade $S_{13,13}$, onde em verde destacam-se as colunas da matriz e em azul destacam-se as linhas. As posições $S_{3,8}$ e $S_{8,3}$ da matriz armazenam o grau de similaridade entre as sequências 3 e 8. Vale ressaltar que existe a possibilidade da matriz de similaridade S ser assimétrica. Ou seja, dados duas sequências *i* e *j* é possível obter o seguinte resultado: S[*i*, *j*] <> S[*j*, *i*]. O grau de similaridade entre as sequências *i* e *j*, comparando inicialmente *i* com *j* e posteriormente *j* com *i*, pode apresentar resultados diferentes por que as regras de edição para o alinhamento entre sequências ilustradas na sessão 4.1.1 e 4.1.2 não garantem que os *gaps*, ou

espaços, sejam inseridos nos mesmos lugares durante o alinhamento das sequencias.

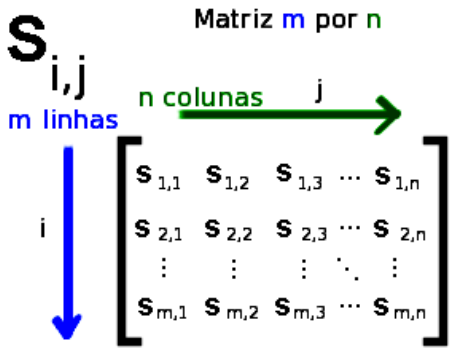


Figura 25: Matriz Similaridade $S_{i,j}$ que armazena o grau de similaridade entre as sequências da tabela SIMILARIDADE (m, linhas) e as sequências da ferramenta BLAST (n, colunas).

Para solucionar esse problema criou-se um programa que verifica a simetria da matriz de similaridade S. Se durante sua execução o programa constatar que a matriz de similaridade é assimétrica ele força a simetria da matriz, utilizando o maior grau de similaridade entre as sequencias onde a assimetria foi constatada. O código fonte do programa que verifica a simetria da matriz de similaridade encontra-se nos apêndices D.

i/j	1	2	3	4	5	6	7	8	9	10	11	12	13
1	**0**	0	95	89	0	93	100	0	100	83	94	0	92
2	0	**0**	87	100	92	100	80	94	0	0	0	87	0
3	95	87	**0**	100	88	0	100	**84**	0	0	0	84	0
4	89	100	100	**0**	100	95	0	0	89	90	95	100	95
5	0	92	88	100	**0**	100	81	92	0	100	0	100	100
6	93	100	0	95	100	**0**	0	0	92	87	84	100	85
7	100	80	100	0	81	0	**0**	91	0	100	100	100	100
8	0	94	**84**	0	92	0	91	**0**	0	0	100	90	0
9	100	0	0	89	0	92	0	0	**0**	89	94	0	100
10	83	0	0	90	100	87	100	0	89	**0**	85	95	85
11	94	0	0	95	0	84	100	100	94	85	**0**	0	89
12	0	87	84	100	100	100	100	90	0	95	0	**0**	100
13	92	0	0	95	100	85	100	0	100	85	89	100	**0**

Figura 26: Exemplo de matriz de Similaridade $S_{i,j}$. Para facilitar o entendimento, destacam-se em verde as colunas e em azul as linhas. Em vermelho destaca-se o grau de similaridade entre as sequências 3 e 8 nas posição $S_{3,8}$ e $S_{8,3}$.

Com base na matriz de similaridade S, particularizou-se a relação entre as sequências criando uma matriz de adjacência para cada grau de similaridade. Ou seja, dado um grau de similaridade (ex.: 85) a matriz de adjacência A seria preenchida seguindo a regra A(i,j) = 1 se existe entre as sequências i e j um grau de similaridade maior ou igual a 85. Caso contrário a posição V(i,j) da matriz de V seria preenchida com o valor 0. A figura 27 ilustra a matriz de adjacência para similaridade maior ou igual a 85% construída a partir da matriz ilustrada na figura 26.

i/j	1	2	3	4	5	6	7	8	9	10	11	12	13
1	**0**	0	1	1	0	1	1	0	1	0	1	0	1
2	0	**0**	1	1	1	1	0	1	0	0	0	1	0
3	1	1	**0**	1	1	0	1	0	0	0	0	0	0
4	1	1	1	**0**	1	1	0	0	1	1	1	1	1
5	0	1	1	1	**0**	1	0	1	0	1	0	1	1
6	1	1	0	1	1	**0**	0	0	1	1	0	1	1
7	1	0	1	0	0	0	**0**	1	0	1	1	1	1
8	0	1	0	0	1	0	1	**0**	0	0	1	1	0
9	1	0	0	1	0	1	0	0	**0**	1	1	0	1
10	0	0	0	1	1	1	1	0	1	**0**	1	1	1
11	1	0	0	1	0	0	1	1	1	1	**0**	0	1
12	0	1	0	1	1	1	1	1	0	1	0	**0**	1
13	1	0	0	1	1	1	1	0	1	1	1	1	**0**

Figura 27: Exemplo de matriz de adjacência $A_{i,j}$ para similaridade maior ou igual a 85%.

As matrizes de adjacência foram processadas e a partir delas as Redes Complexas foram criadas. Os índices das Redes foram dispostos em gráficos e analisados utilizando os seguintes índices: Caminho mínimo médio, Coeficiente de aglomeração, Distribuição de graus, Assortatividade e *Betweenness*. Para visualizar as Redes Complexas, as matrizes de adjacência foram exportadas para o formato (.net) e processadas pela ferramenta PAJEK (BATAGELJ, 2007). A figura 28 ilustra um arquivo com formato .net e na figura 29 uma Rede Complexa criada a partir do arquivo ilustrado na figura 28. Em vermelho destacam-se os vértices e em azul as arestas que estabelecem as relações entre os vértices.

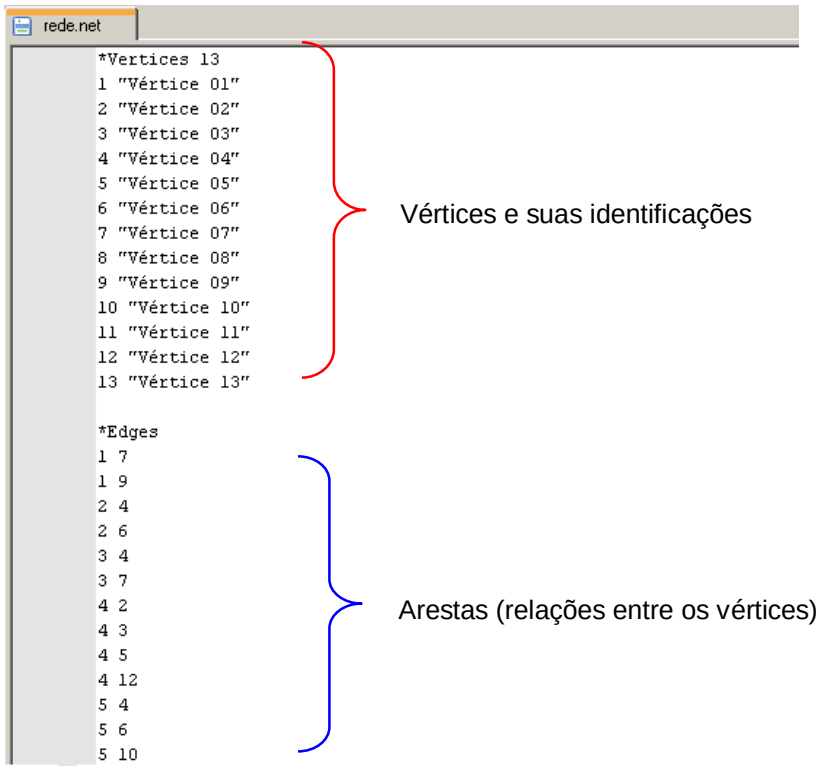


Figura 28: Formato do arquivo .net utilizado pelo PAJEK (BATAGELJ, 2007) para criar as Redes Complexas.

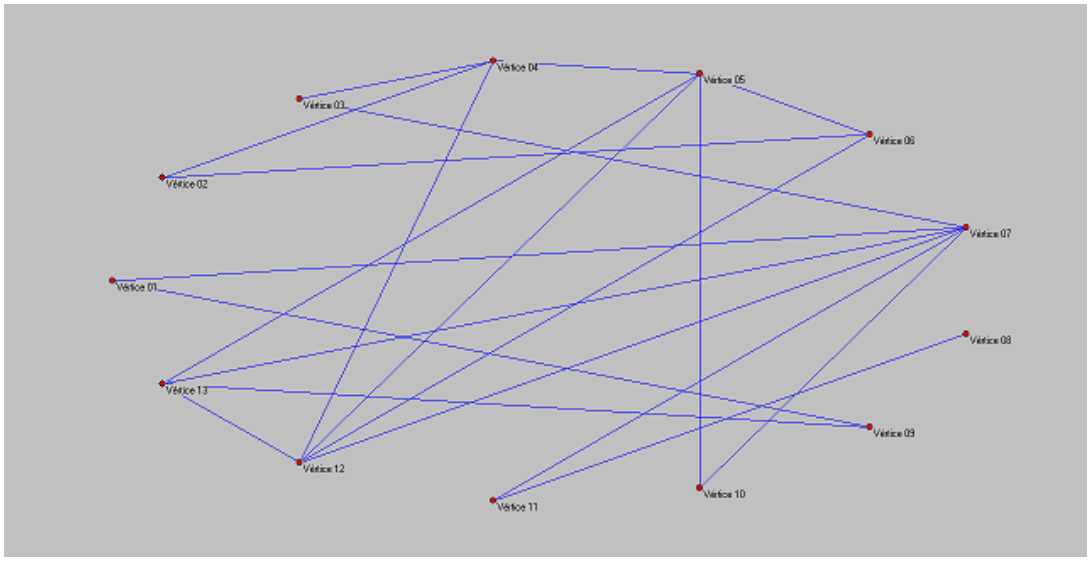


Figura 29: Rede complexa gerada pelo PAJEK (BATAGELJ, 2007) utilizando o arquivo .net ilustrado na figura 25.

O código fonte do script PERL e do programa escrito em C utilizado para construir as matrizes de adjacência encontram-se nos apêndices E.

5.4. Análise Filogenética

Os passos sequenciais utilizados para a análise filogenética das sequências protéicas da sintase da quitina são sintetizados no fluxograma mostrado na Figura 30.

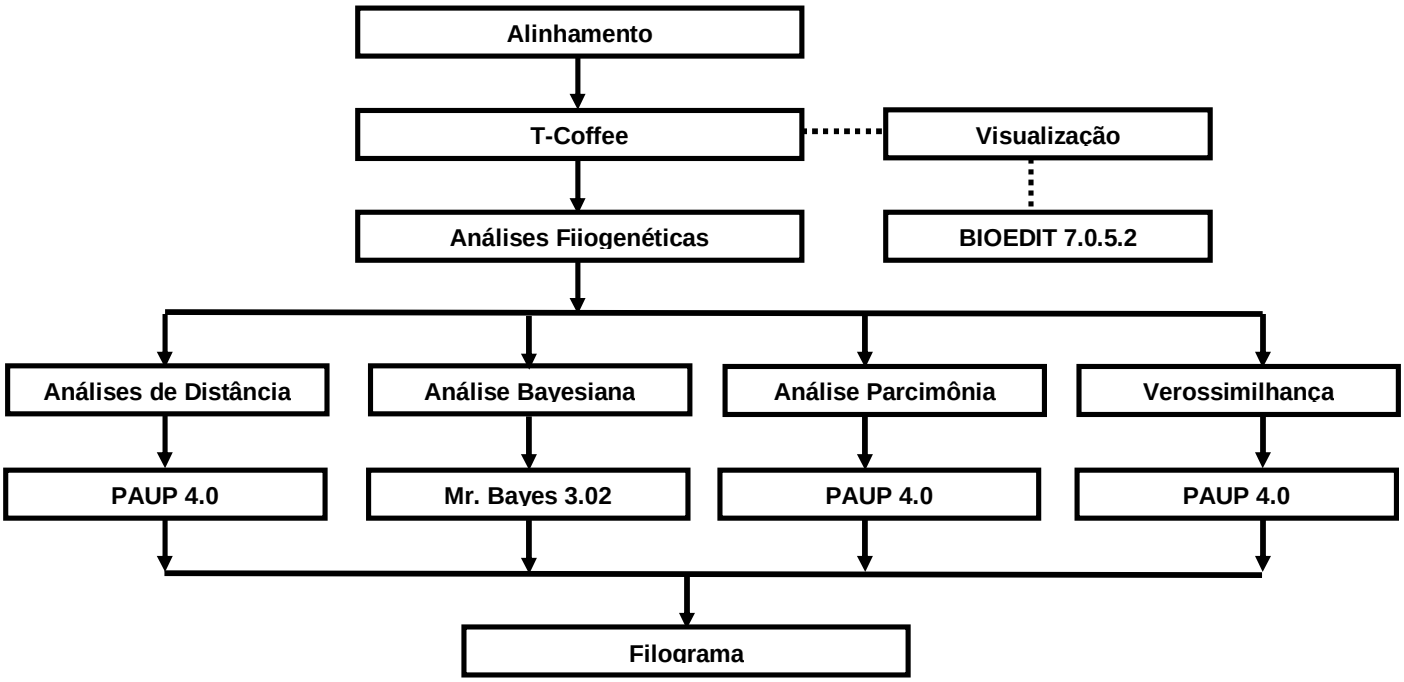


Figura 30: Fluxograma que ilustra os passos sequenciais utilizados para as análises filogenéticas das sequências protéicas da sintase de quitina.

As análises filogenéticas foram realizadas a partir de sequências protéicas completas e parciais, que sintetizam a enzima sintase da quitina em fungos basidiomicetos armazenadas no Banco de Dados. Todas as sequências foram alinhadas no programa T-COFFEE (NOTREDAME, C; HIGGINS, D; HERINGA, 2000). Os resultados desses alinhamentos foram lidos pelo programa *BIOEDIT* 7.0.5.2 (HALL, 1999) e salvos em formato *NEXUS* para serem utilizados pelos programas de filogenia.

A análise de distância, parcimônia e verossimilhança foram realizadas pelo programa PAUP 4.0 (SWOFFORD, 1998). Foram realizadas buscas heurísticas com 1000 pseudoreplicações de *bootstrap*, utilizando-se o algoritmo TBR. Foram obtidos os consensos de maioria das árvores geradas; os dendogramas obtidos

foram salvos e editados pelo programa Treeview 1.6.6 (PAGE, 1996), gerando posteriormente figuras no formato emf (*enhanced metafile*).

Antes da realização das análises filogenéticas bayesianas pelo *MRBAYES* 3.1.2 (RONQUIST; HUELSENBECK, 2003), foram selecionados os modelos evolutivos para cada uma das sequências através do próprio *MRBAYES*. Após isso, os mesmos arquivos de alinhamento utilizados nas análises tradicionais foram configurados manualmente com os modelos de evolução definidos, e submetidos ao *MrBayes*. As análises bayesianas foram configuradas para utilizar os seguintes parâmetros: frequência de amostragem igual a 100, quatro cadeias de aquecimento (três aquecidas e uma fria), valor da parada de aquecimento das cadeias igual a 0,2 e 1.000.000 de gerações, estas configurações tem o objetivo de estabilizar a entrada de dados. As escolhas dos grupos externos seguiram os mesmos critérios das análises tradicionais. Após o final de cada análise foram gerados arquivos no formato *Excel* (xls), contendo os dados de cada replicação. Com esses dados foram gerados gráficos que indicaram os pontos de estabilização das cadeias aquecidas de cada análise e, a partir desses pontos puderam-se sumarizar as estatísticas e obter árvores com os valores de credibilidade de cada grupo de indivíduos. Todas as árvores foram salvas pelo *MrBayes* no formato Treeview e, posteriormente, salvas como figuras no formato emf. Para a análise dos resultados, utilizou-se o Consenso da Maioria das árvores geradas. O Consenso da Maioria é uma forma de sumarizar em uma só árvore o resultado de todas as árvores geradas na análise bayesiana.

5.5. Identificação e Caracterização de Domínios Conservados

A análise de domínios e regiões conservadas da provável proteína foi realizada utilizando-se a busca pelo BLASTp (http://blast.ncbi.nlm.nih.gov/Blast.cgi). Os resultados foram confrontados no InterPro que integra diferentes bases de dados proteômicos, como o UniProt (integra informações sobre proteínas contidas no Swiss-Prot, TrEMBL e pIR), ProSite (base de dados de famílias e domínios de proteínas), Pfam (dados de

múltiplas sequências alinhadas) e SMART (identificação e anotação de domínios e arquiteturas de domínios de famílias protéicas).

6. Resultados e Discussão

Nesta seção são apresentados e discutidos os resultados obtidos nas análises das redes complexas, filogenia, a comparação entre a análise de redes complexas e as análises filogenéticas e a identificação e análise dos domínios conservados das sequências protéicas de sintase da quitina em fungos basidiomicetos. As sequências foram divididas em três grupos: sequências completas, sequências parciais e sequências totais. Este último grupo é composto pela junção das sequências completas e parciais.

6.1. Redes Complexas

Para melhor entendimento, os resultados das Análises das Redes Complexas foram divididos em três partes, seguindo as classificações aplicadas às sequências estudadas. Somente o coeficiente de aglomeração médio e o caminho mínimo médio serão estudados porque essas são as propriedades que melhor representam o fenômeno estudado, ou seja, eles permitem estudar de forma direta a formação de grupos. O objetivo desse experimento é definir o limiar crítico das redes. Este valor definirá o dendrograma que será utilizado na comparação com as árvores filogenéticas geradas pelos métodos tradicionais de análise filogenética. Deve-se lembrar que a propriedade *betweenness* é analisada a cada grau de similaridade entre as sequências protéicas. Todos os índices (coeficiente de aglomeração médio, caminho mínimo médio e *betweenness*) foram calculados a partir do programa MADCHAR (ANDRADE et al, 2006), um programa desenvolvido em FORTRAN pelo FESC, grupo de Física e Estatística Computacional da UFBA.

6.1.1 Sequências Completas de Proteínas (conjunto total de N=39)

Conforme ilustrado na figura 31 a propriedade de coeficiente de aglomeração médio apresentou um comportamento decrescente até atingir a

marca de 43% de similaridade. A partir deste ponto, ela assumiu um comportamento crescente até atingir 47% de similaridade e muda de comportamento aos 48% e 49% de similaridade. A partir desse ponto, 49%, o coeficiente de aglomeração médio se comportou de forma decrescente, mostrando três pontos de estabilidade entre 67% e 72%, 73% e 88% e 89% e 100% de similaridade.

A propriedade de caminho mínimo médio apresentou um comportamento crescente até atingir um pico de 47% de similaridade. Depois desse ponto, observa-se uma grande queda na leitura dessa propriedade nos pontos 48% e 49% de similaridade. Após essa grande mudança de comportamento, é possível notar pontos de estabilidade de 50% a 55%, 56% a 63% e 64% a 100% de similaridade.

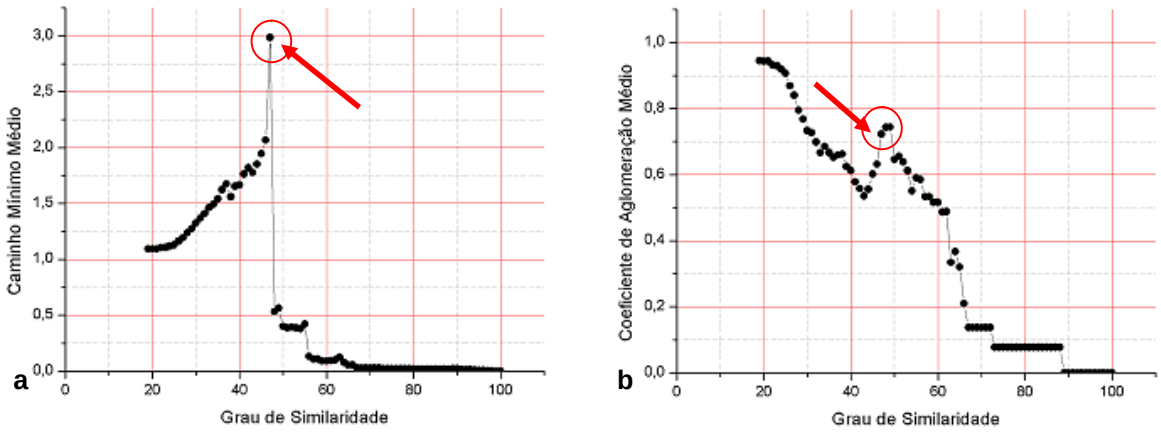


Figura 31: Resultados das análises dos índices das redes complexas das sequências completas de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) caminho mínimo médio; gráfico b) coeficiente de aglomeração médio. Destacam-se em vermelho os pontos com 47% de similaridade.

A leitura dos dendogramas gerados a partir da análise de *betweenness* (NEWMAN, 2004) confirmam os dados ilustrados pela propriedade de coeficiente de aglomeração médio e caminho mínimo médio. A figura 32 ilustra a formação de *clusters* depois de sucessivas eliminações de arestas para os graus de 40% e 47% de similaridade. A figura 31b apresenta quatro *clusters* bem definidos. Esses agrupamentos fragmentam-se à medida que o grau de similaridade se aproxima de 100%.

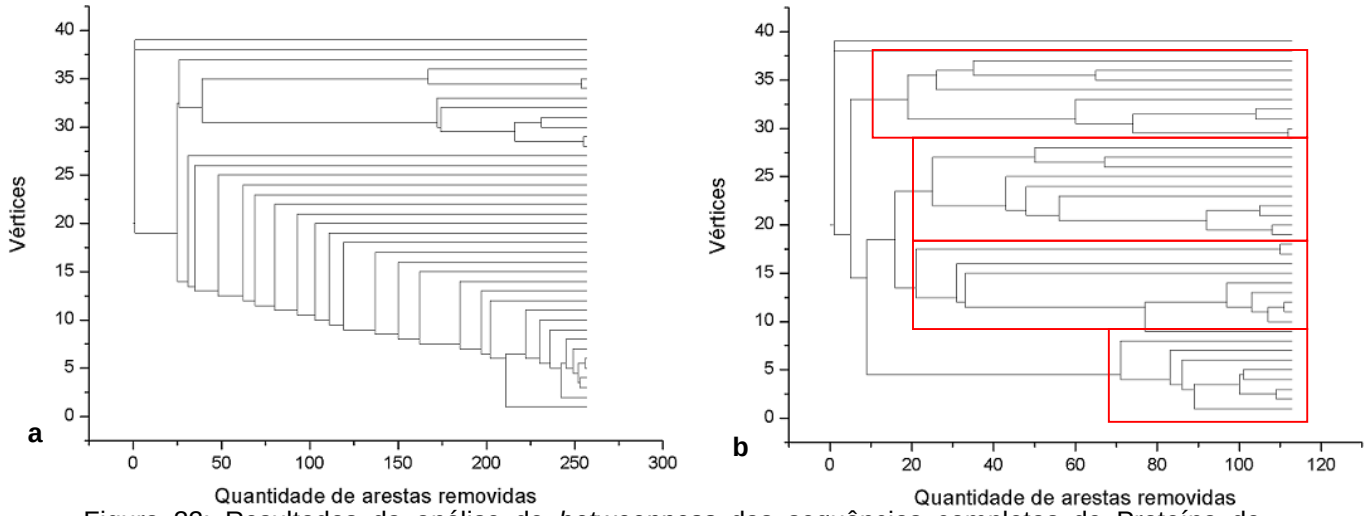


Figura 32: Resultados da análise de *betweenness* das sequências completas de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) ilustra o dendograma para similaridade igual a 40%; e o gráfico b) ilustra o dendograma para similaridade igual a 47%. Em vermelho destacam-se os quatro grupos inicialmente identificados

Para dar uma visão mais concreta dos grupos, as redes foram plotadas utilizando o *software* pajek (BATAGELJ, 2007) (Figura 33), onde é possível notar uma redução significativa no número de arestas nas figuras 33a, 33b e 33c, fato que possibilita a formação de grupos. As matrizes de cores foram utilizadas para obter uma visão mais minuciosa do limiar crítico e das comunidades (Figura 32).

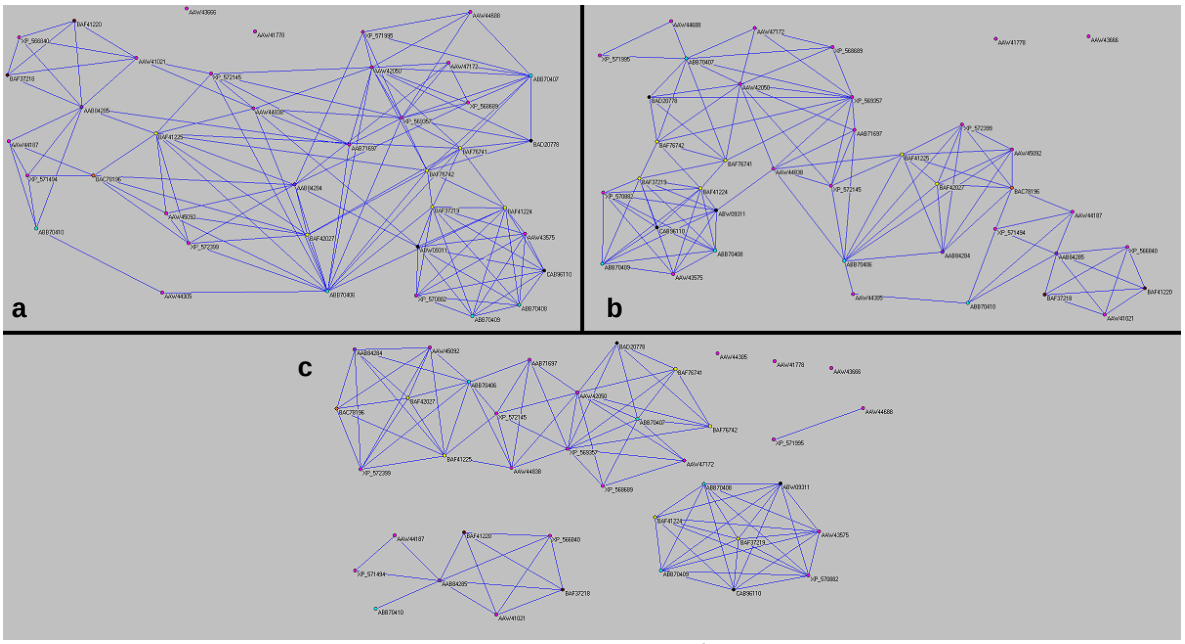


Figura 33: Redes das sequências completas de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) a rede para similaridade igual a 46%; o gráfico b) ilustra a rede para similaridade igual a 47%; e o gráfico c) ilustra a rede para similaridade igual a 48%.

Analisando a figura 34 é possível notar a diferença entre as matrizes e perceber uma mudança de comportamento, ilustrada nas figuras 34a e 34b. Porém, a matriz ilustrada na figura 34c é completamente diferente e retrata uma mudança drástica de comportamento entre os elementos da rede.

Para confirmar as observações feitas nas matrizes de cores da figura 34 foi utilizada a distância euclidiana entre as matrizes de vizinhança de cada rede, figura 35. As redes consecutivas foram comparadas duas a duas, e a maior distância entre matrizes de vizinhança determina o limiar crítico.

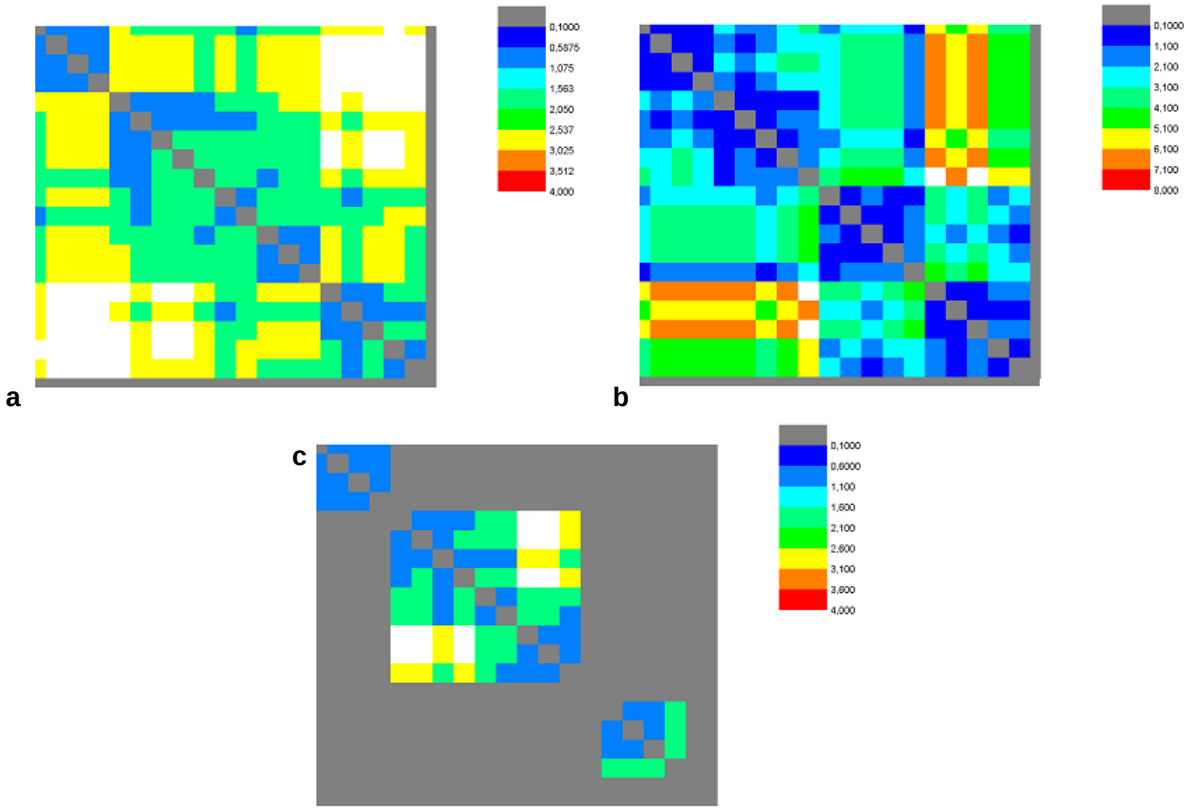


Figura 34: Matriz de cores das sequências completas de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) ilustra a matriz para similaridade igual a 46%; o gráfico b) ilustra a matriz para similaridade igual a 47%; e o gráfico c) ilustra a matriz para similaridade igual a 48%.

Depois de efetuar todas as análises sobre o conjunto de sequências completas de sintase da quitina de fungos basidiomicetos ficou claro que o limiar crítico desse grupo de sequências é 47% de similaridade.

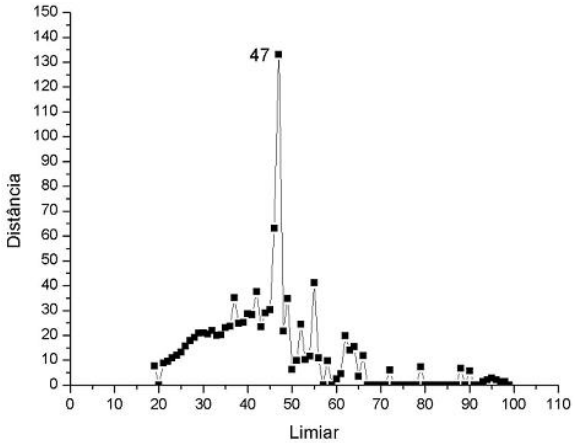


Figura 35: Distância euclidiana entre as matrizes de vizinhança de limiares consecutivos em função do limiar das sequências completas de proteína de sintase da quitina de fungos basidiomicetos.

6.1.2 Sequências Parciais (conjunto de N = 191) e Totais (conjunto N = 230) de Proteínas

Os grupos de sequências parciais e totais apresentaram resultados extremamente próximos, conforme ilustram as figuras 36 e 37. Devido a isso, é possível dizer que, para esse experimento, a adição das 39 sequências completas ao grupo de sequências parciais não modificou o comportamento do grupo de sequências parciais.

Nas figuras 36 e 37 a propriedade caminho mínimo médio assumiu um comportamento crescente até atingir 59% de similaridade. Depois disso é possível notar uma faixa de instabilidade entre 60% e 63% seguida de uma queda brusca para 64% e 65% de similaridade. A partir de 66% o caminho mínimo médio apresentou um comportamento decrescente com faixas de estabilidade de 66% a 68%, 69% a 71%, 72% a 73% e 74% a 100% de similaridade.

O coeficiente de aglomeração médio também apresentou comportamento decrescente do início até o fim de sua leitura. A mudança de comportamento foi observada entre os pontos 78% e 80% de similaridade. Neste intervalo ocorre uma queda acentuada e esse comportamento continua até atingir 100% de similaridade.

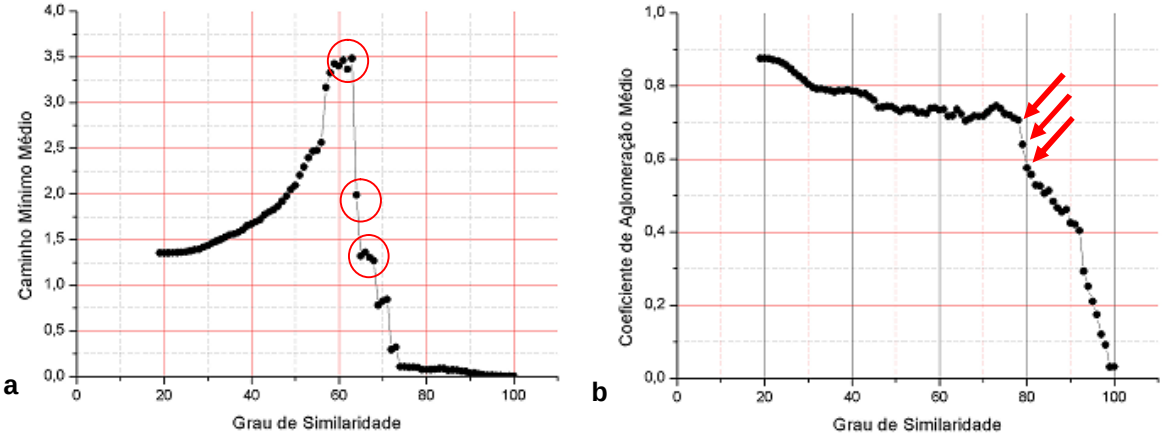


Figura 36: Resultados das análises dos índices das redes complexas das sequências parciais de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) caminho mínimo médio e o gráfico b) coeficiente de aglomeração médio. Em vermelho destacam-se os pontos de mudança de comportamento.

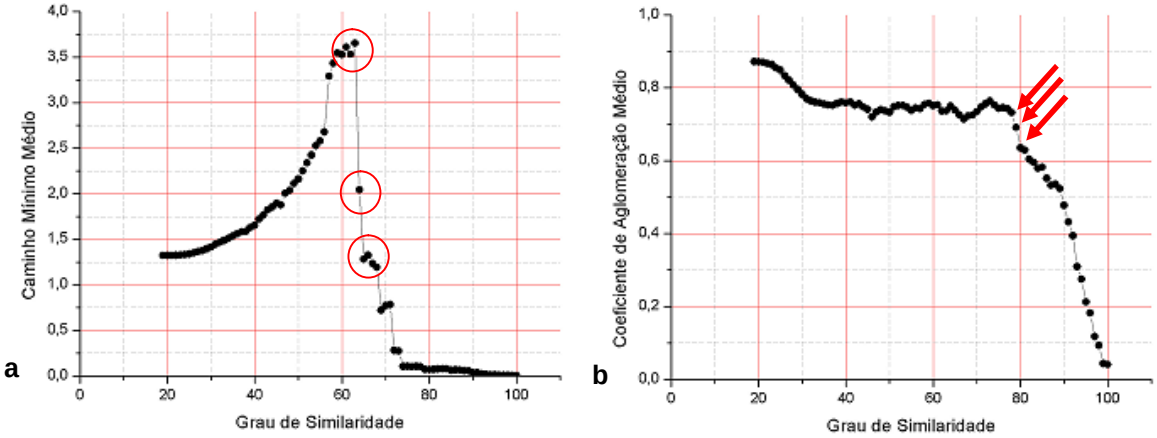


Figura 37: Resultados das análises dos índices das redes complexas de todas sequências de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) caminho mínimo médio e o gráfico b) coeficiente de aglomeração médio. Em vermelho destacam-se os pontos de mudança de comportamento.

Devido à transição brusca ilustrada na figura 36a e na figura 37a o intervalo entre 60% e 65% de similaridade foi escolhido para identificar agrupamentos nos dendrogramas gerados pela análise de *betweenness* (NEWMAN, 2004) e nas matrizes de cores. Porém a transição entre os pontos escolhidos para análise foi muito suave e devido a isso não foi possível extrair informações relevantes da análise de *betweenness* (NEWMAN, 2004). Para facilitar a visualização os dendrogramas gerados das sequências parciais e totais foram plotados em uma escala maior no apêndice H e I. Consequentemente, a identificação dos limiares críticos e dos grupos de sequências parciais e totais foi determinada com base na distância euclidiana ilustrada na figura 38 e nas matrizes de cores, na figura 39 e 40.

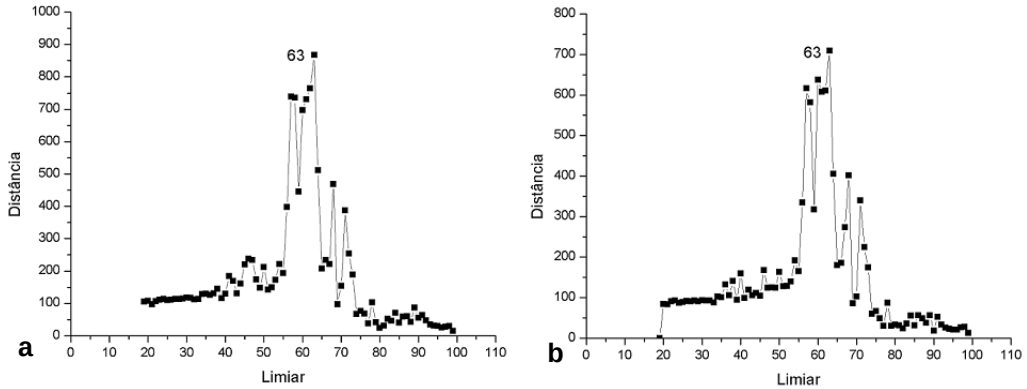


Figura 38: a) distância euclidiana entre as matrizes de vizinhança de limiares consecutivos em função do limiar de todas sequências de proteína de sintase da quitina de fungos basidiomicetos e b) distância euclidiana entre as matrizes de vizinhança de limiares consecutivos em função do limiar das sequências parciais de proteína de sintase da quitina de fungos basidiomicetos.

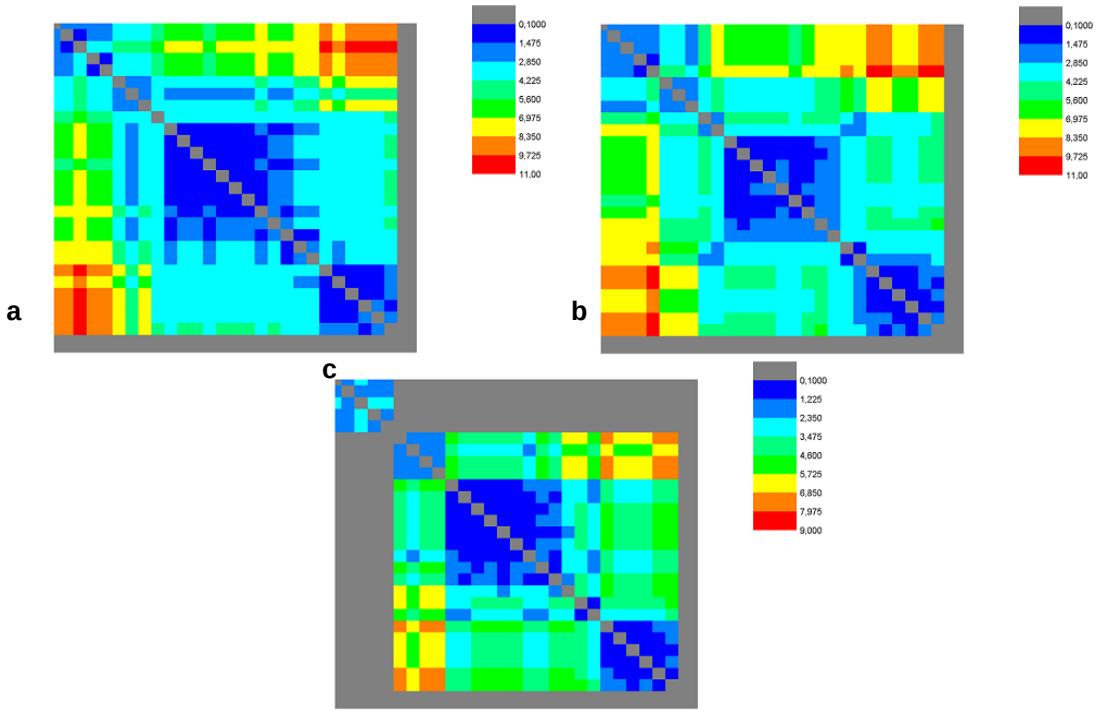


Figura 39: Matriz de cores de todas sequências de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) a matriz para similaridade igual a 62%; o gráfico b) ilustra a matriz para similaridade igual a 63%; e o gráfico c) ilustra a matriz para similaridade igual a 64%.

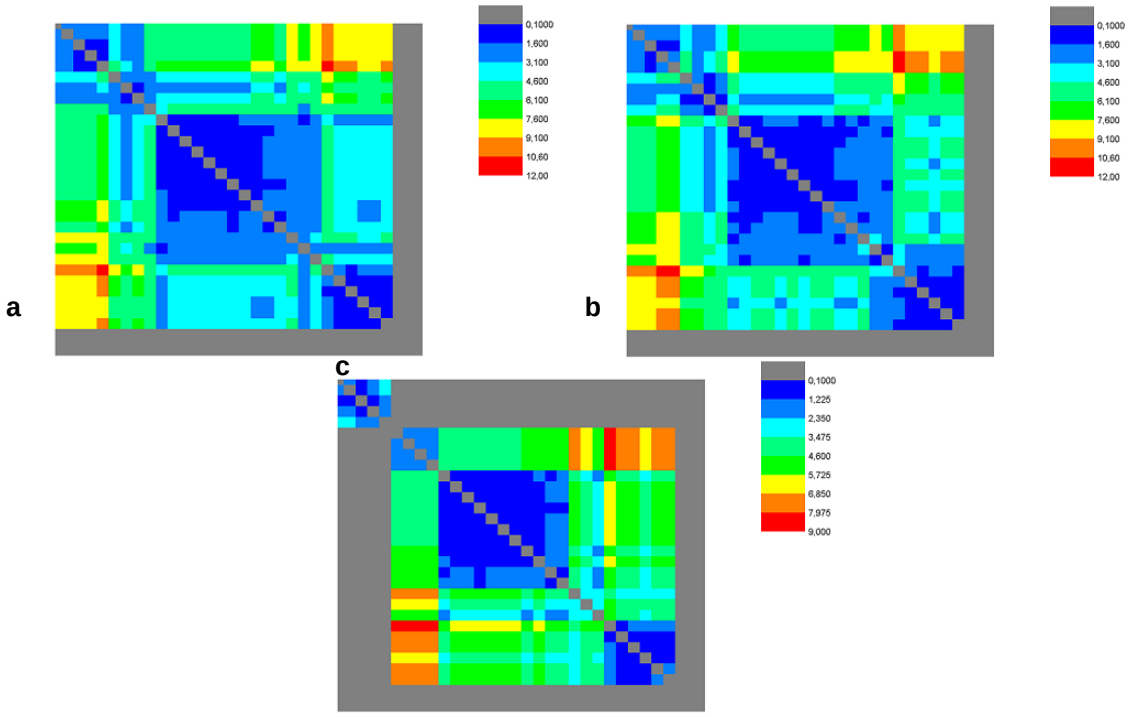


Figura 40: Matriz de cores das sequências parciais de Proteína de Sintase da Quitina de Fungos Basidiomicetos. O gráfico a) a matriz para similaridade igual a 62%; o gráfico b) ilustra a matriz para similaridade igual a 63%; e o gráfico c) ilustra a matriz para similaridade igual a 64%.

Depois de efetuar as analises, o limiar crítico das sequências totais e parciais de proteína de sintase da quitina de fungos basidiomicetos foi definido em 63% de similaridade. O código fonte do programa escrito em Java utilizado para definir a distância euclidiana entre as redes encontra-se nos apêndices F.

6.2. Comparações entre as Redes Complexas e os métodos tradicionais de Filogenia

O objetivo desse experimento foi definir o grau de congruência entre os grupos formados pelos dendogramas das redes complexas e os grupos formados pelas árvores filogenéticas geradas através dos métodos tradicionais de filogenia. Dessa forma, torna-se necessário explicitar o conceito de congruência utilizado nesta pesquisa.

Sejam duas redes p e q associadas a dois grupos distintos; N_{pq} o número de elementos presentes nas duas redes; M_{pq} o número máximo de elementos na mesma comunidade nas duas redes, chama-se de índice de congruência à razão

ilustrada na fórmula 16. Caso o número de comunidades de *p* e *q* sejam distintas, faz-se uma correspondência de duas os mais comunidades da rede *p* e na rede *q*.

$$Gpq = \frac{M_{pq}}{N_{pq}} \qquad (16)$$

A figura 41 ilustra a definição de índice de congruência. Nela é possível observar duas redes. A rede *p* que possui onze (11) comunidades e a rede *q*, que possui seis (6) comunidades. A rede *p* possui 245 elementos, a rede *q*, possui 88 elementos. Os números que dizem respeito às duas redes e que serão utilizados para o cálculo do índice de congruência são: N_{pq} = 44, M_{pq} = 40. A razão entre M_{pq} e N_{pq} é igual à G_{pq} = 40/44 ou 90,9% de congruência.

a)

	q1	q2	q3	q4	q5	q6
p1	0	0	0	0	2	0
p2	0	0	3	1	0	0
p3	0	0	0	0	0	0
p4	0	0	0	0	0	0
p5	0	0	0	3	0	0
p6	0	0	8	1	0	0
p7	0	0	8	0	0	0
p8	2	0	0	0	0	0
p9	0	10	2	0	0	0
p10	0	0	0	0	0	0
p11	0	4	0	0	0	0

b)

Interseção	44

c)

Associação	40
q1p8	2
q2p9p11	14
q3p2p6p7	19
q4p5	3
q5p1	2

d)

Não congruentes	4
q3p9	2
q4p2	1
q4p6	1

e)

Congruência	40/44
Congruência	90,9%

Figura 41: A tabela a) ilustra a interseção entre os elementos das redes. A tabela b) ilustra a quantidade de elementos comuns as duas redes. Nas tabelas c) e d) é possível observar os elementos congruentes e os elementos não congruentes da redes. Na tabela e) o grau de congruência das redes.

Nesse experimento, somente o grupo de sequências completas foi estudado. Essa decisão visa diminuir o ruído que as sequências parciais inserem nas análises devido à grande diferença de tamanho que existe entre as

sequencias completas e parciais. Desta maneira, um grande número de *gaps* é utilizado durante o alinhamento das sequências. O código fonte do programa escrito em Java utilizado para definir o grau de congruência entre os grupos encontra-se no apêndice G.

6.2.1. Resultados das comparações entre as Redes Complexas e os métodos tradicionais de Filogenia

A tabela 02 ilustra os resultados obtidos durante a comparação entre a rede complexa com limiar igual a 47 e os métodos tradicionais de filogenia.

Tabela 02 – Comparação Redes Complexas X Métodos tradicionais de filogenia

Método de filogenia	Percentual
Análise bayesiana	46%
Análise de distância	51%
Análise de parcimônia	33%
Análise de verossimilhança	51%

Para facilitar a visualização dos grupos formados pelos métodos de filogenia tradicional e da rede complexa os dendrogramas gerados foram plotados lado a lado nas figuras 42, 43, 44 e 45. Em vermelho destacam-se os grupos idênticos e em azul os grupos semelhantes. As setas verdes mostram os pontos de corte usados para formação dos grupos utilizados para o cálculo da congruência.

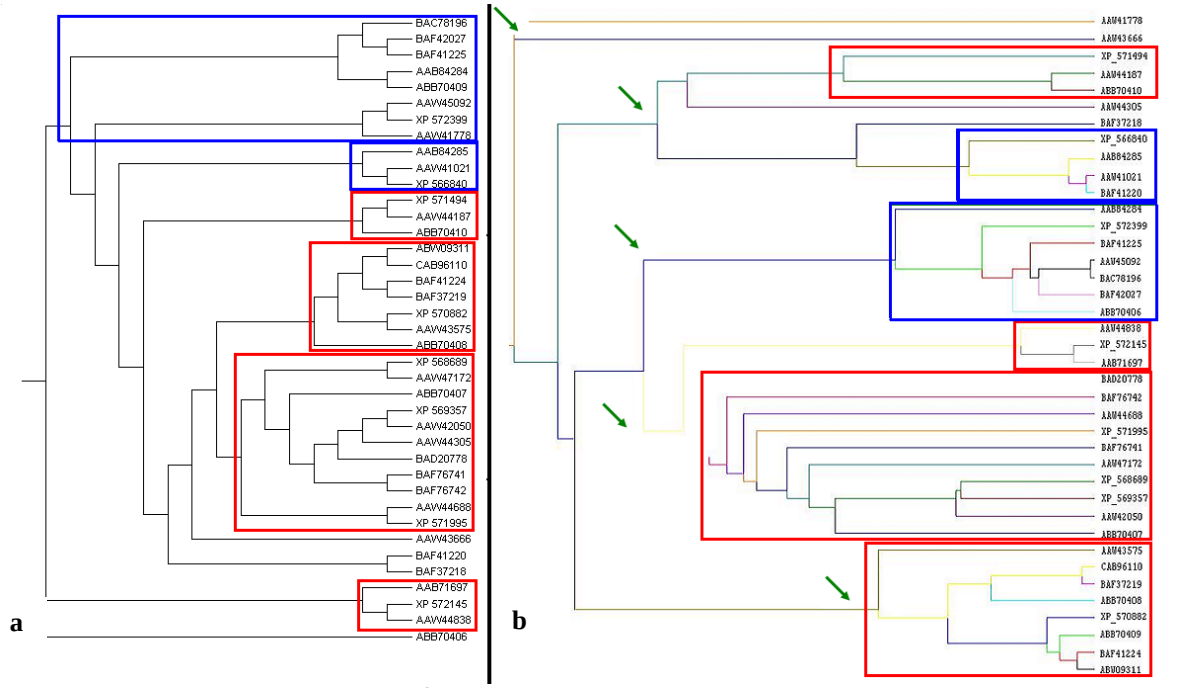


Figura 42: Comparação entre as técnicas tradicionais de filogenia e as redes complexas. Em a) ilustra-se o dendrograma gerado pela análise bayesiana e em b) o dendrograma gerado pelas redes complexas. Em vermelho destacam-se os grupos idênticos e em azul os grupos semelhantes. As setas verdes ilustram os pontos de corte usados para formar os grupos utilizados no cálculo da congruência.

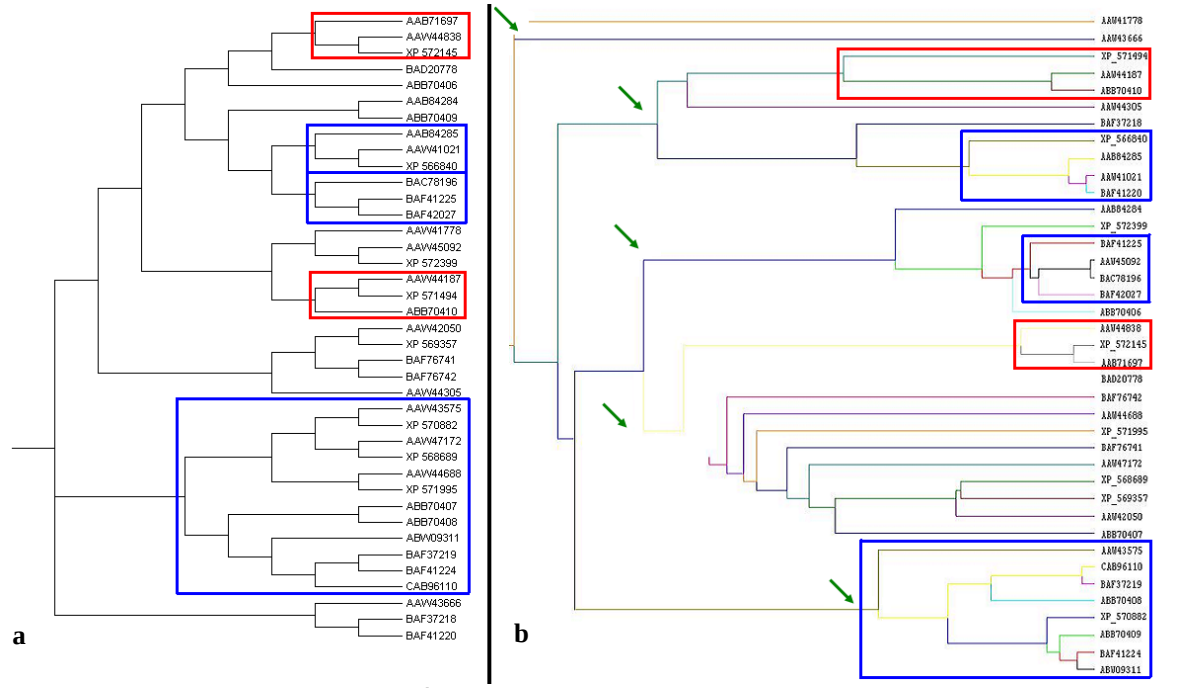


Figura 43: Comparação entre as técnicas tradicionais de filogenia e as redes complexas. Em a) ilustra-se o dendrograma gerado pela análise de distância e em b) o dendrograma gerado pelas redes complexas. Em vermelho destacam-se os grupos idênticos e em azul os grupos semelhantes. As setas verdes ilustram os pontos de corte usados para formar os grupos utilizados no cálculo da congruência.

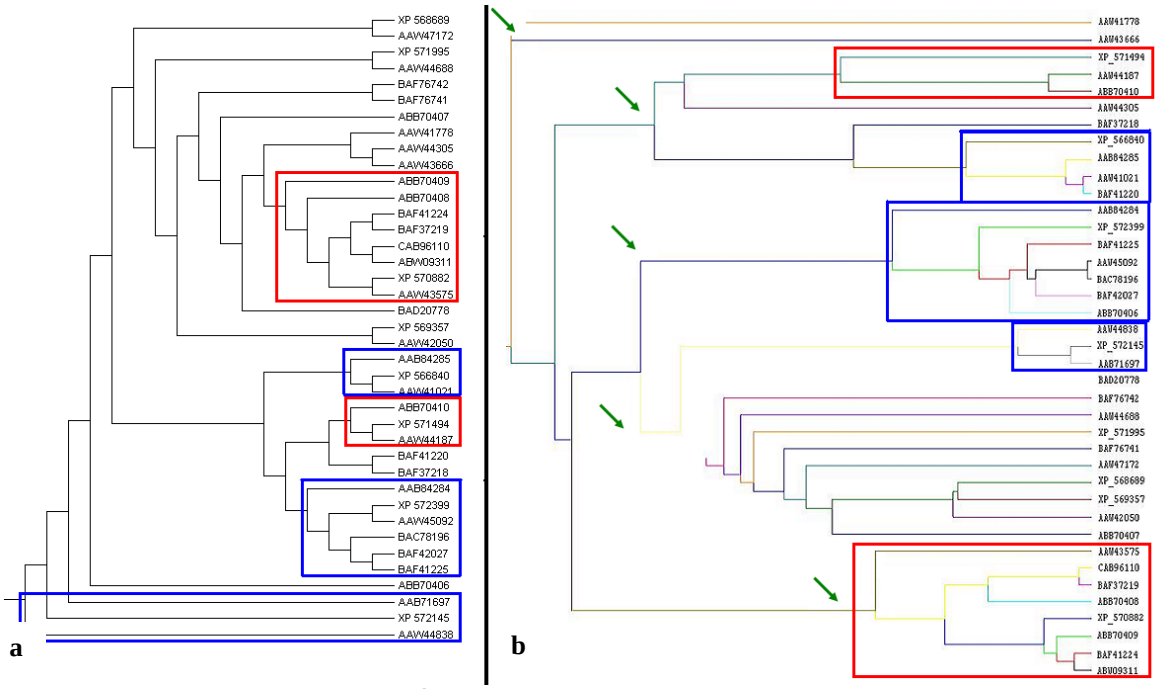


Figura 44: Comparação entre as técnicas tradicionais de filogenia e as redes complexas. Em a) ilustra-se o dendrograma gerado pela análise de parcimônia e em b) o dendograma gerado pelas redes complexas. Em vermelho destacam-se os grupos idênticos e em azul os grupos semelhantes. As setas verdes ilustram os pontos de corte usados para formar os grupos utilizados no cálculo da congruência.

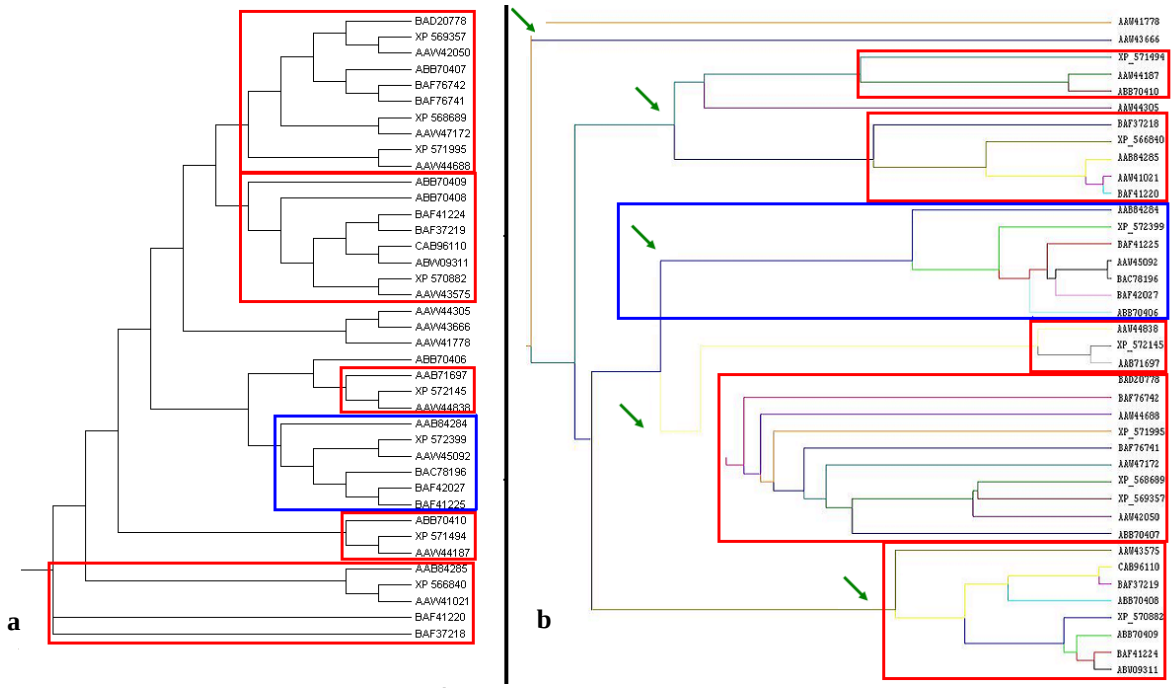


Figura 45: Comparação entre as técnicas tradicionais de filogenia e as redes complexas. Em a) ilustra-se o dendrograma gerado pela análise de verossimilhança e em b) o dendrograma gerado pelas redes complexas. Em vermelho destacam-se os grupos idênticos e em azul os grupos

semelhantes. As setas verdes ilustram os pontos de corte usados para formar os grupos utilizados no cálculo da congruência.

6.2.2. Resultados das comparações entre os métodos tradicionais de Filogenia

A tabela 03 ilustra os resultados obtidos durante a comparação entre as análises bayesiana, distância, parcimônia e verossimilhança.

Tabela 03 – Comparação entre os métodos de filogenia tradicionais

	Bayesiana	**Distância**	**Parcimônia**	**Verossimilhança**
Bayesiana		33%	17%	66%
Distância	33%		35%	58%
Parcimônia	17%	35%		58%
Verossimilhança	66%	58%	58%	

Embora os métodos tenham bases teóricas diferentes, os resultados apresentados na tabela 03 possibilitam estabelecer um grau de qualidade aos resultados obtidos pelas redes complexas. A tabela 04 foi construída ordenando os métodos de filogenia tradicional com base na média da congruência apresentado na tabela 03.

Tabela 04 – Média de congruência dos métodos tradicionais de filogenia

Ordem	**Método**	**Congruência**
1°	Verossimilhança	60%
2°	Distância	42%
3°	Bayesiana	38%
4°	Parcimônia	36%

Na tabela 05 foram adicionados os resultados obtidos pelo dendograma obtido pelo método de redes complexas no limiar crítico (47%). É possível observar na tabela 05 que o dendograma apresentou um grau de congruência acima da média de três dos quatro métodos tradicionais de filogenia, ficando atrás somente da análise de verossimilhança.

Esse resultado é animador devido à possibilidade que ele traz. Os métodos de filogenia mais utilizados pela comunidade científica, análise bayesiana e

análise de verossimilhança, utilizam modelos matemáticos computacionalmente pesados, e devido a isso, trabalhar com um número grande de sequências nessas análises (acima de noventa sequências) torna-se uma tarefa inviável e, em algumas situações, impossível.

Tabela 05 – Média de congruência dos métodos tradicionais de filogenia e o de redes complexas (dendograma em limiar crítico de 47%)

Ordem	Método	Congruência
1°	Verossimilhança	60%
2°	**Redes Complexas**	**45%**
3°	Distância	42%
4°	Bayesiana	38%
5°	Parcimônia	36%

Para execução das análises filogenéticas utilizando os métodos tradicionais foi utilizado o seguinte sistema computacional: processador Celerom 1,6 Ghz, 1GB de memória RAM e Sistema Operacional Windows XP Professional com o *service pack* 2. A tabela 06 ilustra o tempo computacional consumido pelos experimentos descritos na sessão 5.4.

Tabela 6 – Tempo computacional consumido pelos métodos tradicionais de filogenia. Os resultados aproximados foram arredondados para cima.

Método de Filogenia	Sequencias Completas	Sequencias Parciais	Sequencias Totais
Bayesiana	Aproximadamente 74 horas	Mais de 4 meses	Experimento não executado
Verossimilhança	Aproximadamente 53 horas	Mais de 4 meses	Experimento não executado
Distância	Aproximadamente 5 minutos	Aproximadamente 10 minutos	Aproximadamente 15 minutos
Parcimônia	Aproximadamente 5 minutos	Aproximadamente 10 minutos	Aproximadamente 15 minutos

Por outro lado, utilizando as Redes Complexas, é possível trabalhar com um número grande de elementos como trezentas ou mais sequências. Isso possibilita estudos mais abrangentes sobre sistemas biológicos que funcionam a partir da interação de um grande número de proteínas como, por exemplo, as rotas metabólicas (NEWMAN, 2007). Utilizando o mesmo sistema computacional a

metodologia que utiliza as Redes Complexas analisou os três grupos de sequências (sequencias completas, parciais e totais) em 5 horas.

Esses resultados permitem afirmar que as Redes Complexas, além de apresentar resultados similares aos resultados apresentados pelos métodos de filogenia tradicionais, são computacionalmente mais baratas e viáveis, devido ao tempo utilizado para analisar um grupo de 230 sequências protéicas.

6.3. Identificação e caracterização dos Domínios Conservados

Foram identificados um total de 51 domínios conservados nas 230 sequências protéicas (completas e parciais) de sintase da quitina de basidiomicetos depositadas no GenBank. Um total de 221 sequências (96%) apresenta um domínio conservado em comum, CD04190, (Chitin_synth_C). Ele corresponde à porção C-terminal da sintase da quitina. Entre as nove sequências que não apresentam o domínio conservado CD04190, duas (2) apresentam somente domínios específicos, duas (2) são completas e cinco (5) correspondem a sequências parciais da região inicial ou mediana e que não apresentam a porção C-terminal da proteína.

Mais de 50% das sequências podem apresentar ainda um ou mais dos seguintes domínios conservados: COG1215 (89%), CD06423 (74%), PFAM01644 (66%), PFAM03142 (63%) e CD06434 (53%).

O COG1215 corresponde a um grande domínio conservado das glicosiltransferases, responsáveis pela biogênese de parede celular. O domínio conservado CD06423 (CESA-like) caracteriza a superfamília da sintase da celulose, que também é uma glicosiltransferase. O domínio conservado PFAM01644 corresponde a uma região que é comumente encontrada em sintases da quitina das classes I, II e III. O PFAM03142 também é característico das sintases da quitina. O domínio conservado CD06434 (GT2_HAS) corresponde as sintases de hialuronano, que também são glicosiltransferases.

Utilizando as informações da base de dados é sugerido que a identificação *in silico* de uma proteína como sendo a sintase da quitina seja feita,

principalmente, com base no domínio conservado CD04190 e posteriormente combinando os outros cinco domínios que ocorreram em mais de 50% das sequências.

Sugere-se também que inibidores genéricos de sintase da quitina em fungos basidiomicetos sejam sintetizados tendo como alvo molecular o domínio CD04190. Além disso, foram identificados 12 domínios exclusivos que ocorrem nas sintases da quitina de determinados basidiomicetos. Inibidores específicos para esses fungos, *Agaricus bisporus, Amanita verna, Coprinopsis cinerea, Cryptococcus neoformans* e *Malassezia slooffiae*, devem ser construídos tendo como alvo molecular os domínios (BAA34380, BAD06751, CD04192, CD05009, CD06103, COG0449, COG2222, EAU84753, PFAM01380, PRK00331, PTZ00295, TIGR01135).

7. CONCLUSÕES

Neste trabalho foi realizada uma análise computacional em larga escala das sequências protéicas de sintases da quitina de Basidiomycota. Construiu-se um banco de dados relacional contendo todas as sequências protéicas, completas e parciais, de sintases da quitina de Basidiomycota armazenadas no NCBI (NCBI, 2007) até o dia 11/09/2009. Posteriormente, utilizando as sequências protéicas armazenadas no banco de dados, as Redes Complexas foram criadas e analisadas. As sequências protéicas foram divididas em três grupos (sequências completas, parciais, e totais) e submetidas às quatro análises filogenéticas tradicionais (Parcimônia, Distância, Verossimilhança e Bayesiana).

Durante a comparação dos dados gerados pelas Redes Complexas e pelos métodos tradicionais de análise filogenetica foi observado um resultado animador. A metodologia que utiliza as Redes Complexas para definir as relações de parentesco entre individuos apresentou a mesma aficiencia dos metodos tradicionais de filogenia. Comparando os grupos de organismos formados pelas Redes Complexas com os grupos formados pelos métodos tradicionais de filogenia foi encontrado um grau de congruência maior do que os métodos de Parcimônia, Distancia e análise Bayesiana, ficando atrás somente da análise de Verossimilhança. Tendo como base os resultados apresentados nas tabelas 02 e 03 é possível afirmar que as Redes Complexas, aliadas aos conceitos de análise de similaridade, são um método matemático alternativo que reconstrói bem as relações de parentesco entre indivíduos.

Além disso, as Redes Complexas mostraram mais eficácia durante a definição das relações de parentesco entre individuos. O custo computacional dos métodos tradicionais de análise filogenética mais utilizados pela comunidade acadêmica, análise Bayesiana e a análise de Verossimilhança, são altos, ver tabela 06, e em algumas situações torna-se impossível utilizar esses métodos. A metodologia que utiliza as Redes Complexas analisou em cinco horas o maior grupo de sequências protéicas, enquanto as análises de Verossimilhança e a

análise Bayesiana utilizaram, respectivamente, aproximadamente 53 e 74 horas para analisar o menor grupo de sequências protéicas. Devido a isso, é possível dizer que as Redes Complexas são uma solução mais atraente, visto que é não é preciso fazer um grande investimento em recursos computacionais para realizar análises filogenéticas utilizando um grande número de sequências.

Esta metodologia foi recentemente aplicada e testada com um conjunto grande de dados (todas as sequências de UDP N-acetil glicosamina pirofosforilase de todos os organismos com genoma completo sequenciado) com sucesso (Góes-Neto et al., 2010), ver apêndice L.

Os dominios conservados CD04190 (presente em 93% das sequências) e COG1215 (presente em 89% das sequências) são dois potenciais alvos moleculares caso o objetivo seja construir um inibidor de grande abrangência direcionado para a enzima sintase da quitina. De acordo com as sequências estudadas, 221 fungos basidiomicetos seriam atingidos (e teriam sua síntese de quitina inibida ou interrompida) se um inibidor fosse projetado para atacar a estrutura funcional sintetizada por esses domínios.

Por outro lado, os doze (12) domínios exclusivos possibilitam uma grande especificidade. A definição de compostos que ataquem esses alvos moleculares, possibilitaria a atividade inibitória da síntese de quitina específica para uma determinada espécie de basidiomiceto. Isso é de grande importância por que os organismos que não são fito ou zoopatogênicos e têm importância comercial ou para o ambiente em que eles vivem não seriam lesados.

Com base no trabalho desenvolvido duas vertentes podem ser identificadas como trabalhos futuros dando continuidade a pesquisa em questão. A primeira sugestão é a construção de uma ferramenta a partir dos scripts criados durante essa pesquisa. Isso tornaria a metodologia utilizada neste trabalho mais acessível à comunidade acadêmica. A segunda sugestão é a construção de um ambiente distribuído onde a ferramenta criada na primeira sugestão seria um agente, ou um nó, de um ambiente computacional. Isso possibilitaria a utilização de um número muito maior de sequências devido à escalabilidade que um Sistema Distribuído proporciona.

8. REFERÊNCIAS

ALTSCHUL, S.F et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs**.** *Nuc. Acid. Res.*, v. 25, n.17, p. 3389 – 3402. 1997.

AMABIS, J. M ; MARTHO G. R. Biologia das células. V. 1 Origem da vida – Citologia e Histologia. 2. ed. – São Paulo: Moderna, 2004.

AMARAL, L. A. N; OTTINO, J. M. Complex sytems and networks: challenges and opportunities for chemical and biology engineers. Chem. Engin. Scien. V. 59, p.1653-1666, 2004.

ANDRADE, R. F. S.; Miranda, José G. V. ; LOBÃO, Thierry P. .Neighborhood properties of complex networks. Physical Review. E, Statistical, Nonlinear and Soft Matter Physics, v. 73, p. 046101, 2006

ANDRADE, Bruno S. Modelagem por homologia das DNA E RNA polimerases do plasmídeo mitocondrial de *Moniliophthora Perniciosa* e suas relações filogenéticas com outras polimerases fúngicas e virais. Dissertação de Mestrado, Universidade Estadual de Feira de Santana, Departamento de Ciências Biológicas. Feira de Santana, Bahia, 2008.

ANDRADE, R. F.S. ; Pinho, S. T.R. ; Petit Lobão, T . Identification of community structure in networks using higher order neighborhood concepts. International Journal of Bifurcation and Chaos in Applied Sciences and Engineering, v. 19, p. 2677-2685, 2009.

BAGO, B.; CHAMBERLAND, H.; GOULET; A.; VIERHEILIG, H.; LAFONTAINE, J. G.; PICHE, E. Effect of nikkomycin Z, a chitin synthase inhibitor, on hyphal growth and cell wall structure of two arbuscular-mycorrhizal fungi. Protoplasma, v. 192, p. 80–92, 1996.

BARABÁSI, A. L.; ALBERT, R. Emergence of scaling in random networks. Science. 286,509-512, 1999.

BARABÁSI, A. L.; ALBERT, R. Statistical Mechanics of Complex Networks. The American Physical Society. n.1, v.74, p. 47-97, 2002.

BARABÁSI, A. L.; OLTVAI, Zoltán. Network biology: understanding the cell's functional organization. Nature. v.5, p. 101-113, 2004.

BATAGELJ, Vladimir, MRVAR, Andrej. Pajek: Program for Large Network Analysis. Disponível em: < http://vlado.fmf.uni-lj.si/pub/networks/pajek/>. Acesso em: 06 agosto 2007.

BOWMAN, S. M.; FREE, S. J. The structure and synthesis of the fungal cell wall. BioEssays, v. 28, p. 799-808, 2006.

BRITO, T. Rogério. Alinhamento de Sequências Biológicas, Dissertação de Mestrado, Universidade de São Paulo, Instituto de Matemática e Estatística, São Paulo, 2003.

BROWER, J.E.; ZAR, J.H. Field & laboratory methods for general ecology. 2.ed. Dubuque: Wm. C. Brown Publishers, 1977. 226p.

BULAWA, C. E.; SLATER, M.; CABIB, E.; AU-YOUNG, J.; SBURLATI, A.; ADAIR, W. L.; ROBBINS, P. W. The *Saccharomyces cerevisiae* structural gene for chitin synthase is not required for chitin synthesis *in vivo*. Cell, v. 46, p. 213-225, 1986.

CARAZZOLLE, F. Marcelo. Métodos de alinhamento de sequências biológicas Disponível em: < http://www.lge.ibi.unicamp.br/lgeextensao2008/>. Acesso em: 15 outubro 2008.

CODD, E. J. A Relational Model of Data for Large Shared Data Banks. ACM, v. 13, p. 377-387, 1970

DAYHOFF, M.O., SCHWARTZ, R.M.,ORCUTT, B.C. A model of evolutionary change in proteins. In Dayhoff, M.O. Atlas of protein sequence and structure. Natl.Biomed. Res. Found., v. 5, p. 345-352, 1978.

DEITEL, H. M.; DEITEL, P. J.; NIETON, T. R.; MCPHIE, D. C.; PERL como programar. 2 ed. Bookman, 2002. 900p.

GALVÃO, Viviane M.; Um modelo para a neoplasia utilizando redes complexas. Dissertação de Mestrado, Universidade Federal da Bahia, Instituto de Física. Salvador, Bahia, 2006.

GALVÃO, V. ; Miranda, J. G. V. ; Andrade, R. F. A. ; Andrade Jr, J. S. ; Gallos, L. K. ; Makse, H. A. .Modularity map of the network of human cell differentiation. Proceedings of the National Academy of Sciences of the United States of America, v. 107, p. 5750-5755, 2010.

GARCÍA-RODRIGUEZ, L. J. TRILLA. J. A, CASTRO, C. VALDIVIESO, M. H DURÁN, A., RONCERO, C. FEBS Letters. v.478, p. 84-88, 2000.

GAVIN AC, Aloy P, Grandi P, Krause R, Boesche M, Marzioch M, Rau C, Jensen LJ, Bastuck S, Dümpelfeld B, Edelmann A, Heurtier MA, Hoffman V, Hoefert C, Klein K, Hudak M, Michon AM, Schelder M, Schirle M, Remor M, Rudi T, Hooper S, Bauer A, Bouwmeester T, Casari G, Drewes G, Neubauer G, Rick JM, Kuster B,

Bork P, Russell RB, Superti-Furga G. Proteome survey reveals modularity of the yeast cell machinery. Nature, 440:631-6. (2004).

GIBAS, C., JAMBECK, P. Desenvolvendo Bioinformática. P. 293, Campus, Rio de Janeiro, 2001.

GEORGOPAPADAKOU, N. H., TKACZ, J. S. The fungal cell wall as a drug target. Trends in Microbiology, v. 3, p. 98-104, 1995.

GLASER, L., BROWN, D. H. The synthesis of chitin in cell-free extracts of *Neurospora crassa*. J. Biol. Chem. v228, p. 729-742, 1957.

GÓES-NETO. A.., DINIZ, M. V. C.; SANTOS, L. B. L., Pinho, S. T. R., Miranda, J.G.V., PETIT L. T., Borges, E. P., EL-HANI, C., Andrade, R.F.S. Comparative protein analysis of the chitin metabolic pathway in extant organisms: a complex network approach. Biosystems, v. 101, p. 59-66. 2010. (doi:10.1016/j.biosystems.2010.04.006)

GOLDANI, A; CARVALHO, G, S. Análise de parcimônia de endemismo de cercopídeos neotropicais (Hemiptera, Cercopidae). Revista Brasileira de Entomologia, n. 3, v. 47, p. 437-442, 2003.

GRANTHAM, R. Amino acid difference formula to help explains protein evolution. Science, v. 185, p.862-864, 1974.

HARRISON, C. J.; LANGDALE, J. A. A step by step guide to phylogeny reconstruction. *The Plant Journal*, v. 45, p. 561–572. 2006.

HARTL, Daniel L. A primer of populations genetics. 3.ed. Massachusetts: Sinauer, 1999. 219 p.

HIBBETT, D. S., BINDER, M., BISCHOFF, J. F., BLACKWELL, M., CANNON, P. F., ERIKSSON, O. E., HUHNDORF, S., JAMES, T., KIRK, P. M., LÜCKING, R., LUMBSCH, T., LUTZONI, F., MATHENY, P. B., MCLAUGHLIN, D. J., POWELL, M. J. , REDHEAD, S., SCHOCH, C. L., SPATAFORA, J. W., STALPERS, J. A. , VILGALYS, R., AIME, M. C., APTROOT, A., BAUER, R., BEGEROW, D., BENNY, G. L., CASTLEBURY, L. A., CROUS, P. W., DAI, Y.-C., GAMS, W., GEISER, D. M. , GRIFFITH, G. W., GUEIDAN, C., HAWKSWORTH, D. L., HESTMARK, G., HOSAKA, K. , HUMBER, R. A., HYDE, K., IRONSIDE, J. E., KÕLJALG, U., KURTZMAN, C. P., LARSSON, K.-H., LICHTWARDT, R., LONGCORE, J., MIADLIKOWSKA, J., MILLER, A., MONCALVO, J.-M., MOZLEYSTANDRIDGE, S., OBERWINKLER, F., PARMASTO, E., REEB, V., ROGERS, J. D., ROUX, C., RYVARDEN, L., SAMPAIO, J. P., SCHÜßLER, A., SUGIYAMA, J., THORN, R. G., TIBELL, L., UNTEREINER, W. A., WALKER, C., WANG, Z. WEIR, A., WEIß, M., WHITE, M. M., WINKA, K., YAO, Y.-J., ZHANG, N. A higher-level phylogenetic classification of the Fungi. Mycological Research, v. 111, p. 509-547, 2007.

HALL, T.A. Bioedit: a user-friendly biological sequence alignement editor and analysis program for Windows 95/98/NT. *Nucl. Acids. Symp. Ser*., v. 41, p. 95 – 98. 1999.

HOGENKAMP., D. G. Chitin metabolism in insects: chitin synthases and beta-n-acetylglucosaminidases. Abstract of a dissertation. Manhattan, Kansas, 2006.

HOLDER, M.; LEWIS, P. O. Phylogeny estimation: Traditional and Bayesian approaches. *Nat. Rev. Genet.*, v. 4, p. 275-284. 2003.

HUELSENBECK, J.P.; RONQUIST, F. MRBAYES: Bayesian inference of phylogenetic trees. *Bioinf. App. Not*., v. 17, n. 8, p. 754-755. 2001.

JAMES, T. Y.; KAUFF, F.; SCHOCH, C. L.; MATHENY, P. B.; HOFSTETTER, V.; COX, C. J.; CELIO, G.; GUEIDAN, C.; FRAKER, E.; MIADLIKOWSKA, J.; LUMBSCH, H. T.; RAUHUT, A.; REEB, V.; ARNOLD, A. E.; AMTOFT, A.; STAJICH, J. E.; HOSAKA, K.; SUNG, G. H.; JOHNSON, D.; O'ROURKE, B.; CROCKETT, M.; BINDER, M.; CURTIS, J. M.; SLOT, J. C; Wang, Z.; Wilson, A.W; Schulber, A; Longcore, J.E O'DONNELL, K.; MOZLEY-STANDRIDGE, S.; PORTER, D.; LETCHER, P. M.; POWELL, M. J.; TAYLOR, J. W.; WHITE, M. M.; GRIFFITH, G. W.; DAVIES, D. R.; HUMBER, R. A.; MORTON, J. B.; SUGIYAMA, J.; ROSSMAN, A. Y.; ROGERS, J. D.; PFISTER, D. H.; HEWITT, D.; HANSEN, K.; HAMBLETON, S.; SHOEMAKER, R. A.; KOHLMEYER, J.; VOLKMANN-KOHLMEYER, B.; SPOTTS, R. A.; SERDANI, M.; CROUS, P. W.; HUGHES, K. W.; MATSUURA, K.; LANGER, E.; LANGER, G.; UNTEREINER, W. A.; LUCKING, R.; BUDEL, B.; GEISER, D. M.; APTROOT, A.; DIEDERICH, P.; SCHMITT, I.; SCHULTZ, M.; YAHR, R.; HIBBETT, D. S.; LUTZONI, F.; MCLAUGHLIN, D. J.; SPATAFORA, J. W.; VILGALYS, R. Reconstructing the early evolution of Fungi using a six-gene phylogeny. Nature, v. 443, p. 818-822, 2006.

LAGORCE, A., BERRE-ANTON, V., AGUILAR-USCANGA, B., MARTIN-YKEN, H., DAGKESSAMANSKAIA, A., FRANÇOIS, J. Involvement of GFA1, which encodes glutamine–fructose-6-phosphate amidotransferase, in the activation of the chitin synthesis pathway in response to cell-wall defects in *Saccharomyces cerevisiae*. Eur. J. Biochem. v269, p. 1697-1707, 2002.

LEVIN, J.M., ROBSON, B., GARNIER, J. An algorithm for secondary structure determination in proteins based on sequence similarity. FEBS Lett., v. 205, p. 303-308, 1986

LETONDAL, Catherine. SCHUERER, Katja. Bioperl course. Disponível em: <http://www.pasteur.fr/recherche/unites/sis/formation/bioperl/>. Acesso em: 11 abril 2007.

MAR, J.C.; HARLOW, T.J; RAGAN, M.A. Bayesian and maximum likelihood phylogenetic analyses of protein sequence data under relative branch-length differences and model violation. *BMC Evol. Biol*., v. 5, p. 8. 2005.

MARIANI, A. C. Conceito Básicos da Teoria de Grafos. Disponível em: < http://www.inf.ufsc.br/grafos/definicoes/definicao.html>. Acesso em: 7 novembro 2008.

MARCHLER-BAUER A, ANDERSON JB, CHITSAZ F, DERBYSHIRE MK, DEWEESE-SCOTT C, FONG JH, GEER LY, GEER RC, GONZALES NR, GWADZ M, HE S, HURWITZ DI, JACKSON JD, KE Z, LANCZYCKI CJ, LIEBERT CA, LIU C, LU F, LU S, MARCHLER GH, MULLOKANDOV M, SONG JS, TASNEEM A, THANKI N, YAMASHITA RA, ZHANG D, ZHANG N, BRYANT SH. CDD: specific functional annotation with the Conserved Domain Database. Nucleic Acids Res. 2009 Jan; 37(Database issue): D205-10.

MCLACHLAN, A.D. Test for comparing related amino acid sequences. Cytochrome c and cytochrome c551. J. Mol. Biol., v. 61, p. 409-424. 1971.

MERZENDORFER. H. Insect chitin synthases: a review. J Comp Physiol B. v176, p.1-15, 2006.

MIO, T., YABE, T., ARISAWA, M., YAMADA-OKABE, H. The Eukaryotic UDP-Nacetylglucosamine pyrophosphorylases: gene cloning, protein expression, and catalytic mechanism. The Jornal Biological Chemistry. v. 273, p. 14392-14397, 1998.

NETO, P. O. B. Grafos: teoria, modelos, algoritmos. 4. ed. – São Paulo: Edgard Blücher, 2006.

NCBI, National Center for Biotechnology Information. Disponível em: <http://www.ncbi.nlm.nih.gov>. Acesso em: 02 agosto 2007.

NEWMAN, M. E. J; GIRVAN, M. Finding and evaluating community structure in networks. American Physical Society. v. 69, 2004.

NEWMAN, M. E. J. The structure and function of complex networks. Disponível em: < http://www-personal.umich.edu/~mejn/courses/2004/cscs535/review.pdf />. Acesso em: 15 abril 2007.

NONATO, L. G. Tipos e Estruturas de Dados. Disponível em: < http://www.lcad.icmc.usp.br/~nonato/ED/Grafos/node73.html/>. Acesso em: 07 janeiro 2010.

NOTREDAME,C; HIGGINS, D; HERINGA, J. T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment. JMB, v. 302, p.205-217. 2000

PAGE, R.D. TreeView: an application to display phylogenetic trees on personal computers. *Comput. Appl. Biosci.*, v. 12, p. 357 – 358. 1996.

POLANSKI, A.; KIMMEL, M. Bioinformatics. Berlin: Springer, 2007. 376p.

RAO, J.K.M. New scoring matrix for amino acid residue exchanges based on residue characteristic physical parameters. Int. J. Pept. Prot. Res., v. 29, p.276-281, 1987.

RONCERO, C. The genetic complexity of chitin synthesis in fungi. Current Genétics, v. 41, p. 367–378, 2002.

ROCHA, Eduardo. Módulo de Bioinformática: Alinhamento de Sequencias. Disponível em: < http://wwwabi.snv.jussieu.fr/people/erocha/>. Acesso em: 11 janeiro 2008.

ROCHA, Luis E. C. Structural Evolution of the Brazilian Airport Network. Disponível em: < http://arxiv.org/abs/0804.3081v2>. Acesso em: 03 junho 2008

RONQUIST, F.; HUELSENBECK, J.P. MRBAYES 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*, v. 19, p. 1572 – 1574. 2003.

RUIZ-HERRERA, J., GONZÁLEZ-PRIETO, J. M., RUIZ-MEDRANO, R. Evolution and phylogenetic relationships of chitin synthases from yeast and fungi. FEMS Yaest Research, v.1, p. 247-256, 2002.

SANTOS-FILHO, O.A; ALENCASTRO, R.B. Modelagem de Proteínas por Homologia. Quím. Nov., v. 26, n. 2, p. 253 – 259. 2003.

SCHWARTZ , R.M ; DAYHOFF, M. O. Matrices for detecting distance relationships. Atlas of Protein Sequence and Structure, p. 353-358, 1978.

SILBERSCHATZ, A; HENRY F. K. Database Research Faces the Information Explosion. Bell Laboratories Lucent Technologies Inc. New Jersey, EUA, 1996.

SILBERSCHATZ, A; KORTH, H.; SUDARSHAN, S. Sistemas de Banco de Dados. 5.ed. Rio de Janeiro: Campos, 2006. 808 p.

SPECHT, C. A.; LIU, Y.; ROBBINS, P. W.; BULAWA, C. E.; IARTCHOUK, N. The *chsD and chsE gens of Aspergillus nidulans* and their roles in chitin synthesis. Fungal genetics and Biology, v. 20. p. 153-167, 1996.

SWOFFORD, D. L. *PAUP**: Phylogenetic Analysis Using Parsimony (*and Other Methods), Version 4. Sunderland: Sinauer Associates. 1998.

THOMPSON, JD; HIGGINS, DG; GIBSON; T.J. ClustalW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specifc gap penalities and weight matrix choice. *Nuc. Acid. Res*., v. 22, n. 22, p. 4673–4680. 1994.

ZACHARY, W. W. Journal Anthropol, ed. 33, 1977.

WARD, Richard John. Matrizes para análise de similaridade entre sequências. Disponível em:< http://nugen.uece.br/arquivos/iijornadabio/ >. Acesso em: maio 2009.

WANG, X. F; CHEN, G. Complex network: small-world, scale-free and beyond. IEEE Circuits and Systems Magazine, v. 3, p.6-20, 2003.

STROGATZ, H. S; WATTS, D. J. Collective dynamics of ‘small-world network. Nature. 393, 440-442, 1998.

WATTS, Duncan J. Small worlds: the dynamics of network between order and randommes. Published by Princetn Univerty Press. 1999.

YEAGER, A. R., FINNEY, N. S. The first direct evaluation of the two-active site mechanism for chitin synthase. J. Org. Chem, v69, p. 613-618, 2004.

APÊNDICES

APÊNDICE A – Scripts SQL utilizados na construção do banco de dados.

```
CREATE TABLE SEQUENCIA (
        ID_SEQ              BIGINT NOT NULL AUTO_INCREMENT, -- Chave primaria da tabela
        POSICAO_TOT         INTEGER, -- Posição entre todas sequencias
        POSICAO_DNA         INTEGER, -- Posição entre sequencias de DNA
        POSICAO_PRO         INTEGER, -- Posição entre sequencias de proteínas
        POSICAO_PROC        INTEGER, -- Posição entre sequencias de proteínas completas
        POSICAO_PROP        INTEGER, -- Posição entre sequencias de proteínas parciais
        POSICAO_DNAC        INTEGER, -- Posição entre sequencias de DNA completas
        FAMILIA             INTEGER, -- Código da família
        ORDEM               INTEGER, -- Código da ordem
        CLASSE              INTEGER, -- Código da classe
        SUBCLASSE           INTEGER, -- Código da subclasse
        SUBFILO             INTEGER, -- Código do subfilo
        LOCUS               VARCHAR(50), -- Locus
        DEFINITION          VARCHAR(1000), -- Texto descritivo da sequência
        GI                  VARCHAR(50), -- ID único no GenBank
        KEYWORDS            VARCHAR(1000), -- Palavras chave da sequência no GenBank
        ORGANISM            VARCHAR(1000), -- Classificação taxonômica do organismos
        ORGANISM_NAME VARCHAR(50), -- Nome do organismos que a sequência foi extraída
        ARQ                 VARCHAR(50), -- Nome do arquivo que armazena as sequencias
        ALPHABET            VARCHAR(10), -- Tipo de sequência DNA ou Proteína
        COMPLETE_SEQ        VARCHAR(1), -- Flag que define se a sequência é completa
        SEQ                 LONGTEXT, -- Sequência
        PRIMARY KEY         (ID_SEQ)
);


CREATE TABLE SIMILARIDADE (
    ID_SIMILARIDADE    BIGINT NOT NULL AUTO_INCREMENT, -- Chave primaria da tabela
    COD_SEQ_ALVO       BIGINT, -- Código da sequência que vai ser comparada
    COD_SEQ_SIMILAR BIGINT, -- Código da sequência similar a comparada
    PERCENTUAL         FLOAT(7,4), -- Percentual de similaridade
    SCORE              INT, -- Score da similaridade
    EVALUE             FLOAT(40,30), -- Confiabilidade
    LENGTH             INT, -- Tamanho da sequência
    GAPS               INT, -- Número de espaços inseridos
    FRAC_IDENTICAL     FLOAT(40,30), -- Percentual de identidade
    FRAC_CONSERVED FLOAT(40,30), -- Percentual de conservação
    N                  INT, -- Quantidade de scores máximos utilizado para determinar a
confiabilidade
    MATCHES            INT -- Número de combinações com sucesso
    PRIMARY KEY        (ID_SIMILARIDADE)
);
```

```
DELIMITER //

CREATE PROCEDURE pr_sequencias(IN lo varchar(50), IN df varchar(1000), IN gi
VARCHAR(50), IN ky VARCHAR(1000), IN og VARCHAR(1000), IN seq LONGTEXT, IN ar
VARCHAR(50), IN alf VARCHAR(10))

BEGIN

        insert into SEQUENCIA (LOCUS, DEFINITION, GI, KEYWORDS, ORGANISM, SEQ, ARQ,
        ALPHABET) values (lo, df, gi, ky, og, seq, ar, alf);

END

//DELIMITER;


DELIMITER //

CREATE PROCEDURE pr_similaridade(IN ca varchar(200), IN cs VARCHAR(20), IN si
VARCHAR(20), IN sc VARCHAR(20), IN ev VARCHAR(50), IN le VARCHAR(50), IN gap
VARCHAR(50), IN fri VARCHAR(50), IN frc VARCHAR(50), IN n VARCHAR(50), IN mat
VARCHAR(50), IN nui VARCHAR(50), IN nuc VARCHAR(50))

BEGIN

        IF (NOT EXISTS (SELECT 1 FROM SEQUENCIA, SIMILARIDADE WHERE ID_SEQ = ca
        AND COD_SEQ_SIMILIAR = cs AND ID_SEQ = COD_SEQ_ALVO) ) THEN

                insert into SIMILARIDADE (COD_SEQ_ALVO, COD_SEQ_SIMILIAR,
                PERCENTUAL, SCORE, EVALUE, LENGTH, GAPS, FRAC_IDENTICAL,
                FRAC_CONSERVED, N, MATCHES, NUM_IDENTICAL, NUM_CONSERVED)
                values (ca, cs, si, sc, ev, le, gap, fri, frc, n, mat, nui, nuc);

        END IF;

END
//DELIMITER;
```

APÊNDICE B - Scripts escritos em PERL utilizado para inserir registros na tabela SEQUENCIA.

```
#!/bin/perl

  $diretorio = "./";

  opendir(diretorio, "$diretorio");
  @lista = readdir(diretorio);
  closedir(diretorio);

  foreach $arquivo(@lista){

    if ( substr($arquivo, index($arquivo,"."), length($arquivo) ) eq ".fasta"){

      # processando arquivo
      @args = ("perl","readFile.pl", $arquivo,".");
      system(@args) == 0 or die "system @args failed: $?";

    }

  }
```

```
use Bio::SeqIO; # biblioteca perl de biotecnologia
use DBI; # biblioteca perl de acesso a banco de dados


        $database = "xxx"; # nome do banco de dados
        $host = "xxx"; # nome da maquina
        $usuario = "xxx"; # usuario
        $senha = "xxx"; # senha

        # conectando ao banco de dados
        my $dbh = DBI->connect("DBI:mysql:database=$database;host=$host","$usuario",
        "$senha",{'RaiseError' => 1});

        my $objSeq = Bio::SeqIO->new('-file' => "<$ARGV[0]", '-format' => "genbank" );

        my $fileName = "$ARGV[0]";  # variavel que recebe o nome do arquivo de entrada

        my $cont = 0;  # variavel de controle

        print "\n\n";

        print $fileName, "\n\n";


        while (my $seq = $objSeq->next_seq) {

                my $arquivo = $fileName;
                my $locus = $seq->display_id;
                my $definition = $seq->desc;
```

```
                my $gi = $seq->primary_id;
                my $keywords = $seq->keywords;
                my $alphabet = $seq->alphabet;
                my $sequence = $seq->seq;

                $locus      =~ s/["']/\_/g; # retirando aspas para inserir no bd
                $definition =~ s/["']/\_/g; # retirando aspas para inserir no bd
                $gi         =~ s/["']/\_/g; # retirando aspas para inserir no bd
                $keywords   =~ s/["']/\_/g; # retirando aspas para inserir no bd
                $alphabet   =~ s/["']/\_/g; # retirando aspas para inserir no bd
                $sequence   =~ s/["']/\_/g; # retirando aspas para inserir no bd

                $organism = "";

                for my $objFeatures ($seq->get_SeqFeatures) {
                        if ($objFeatures->primary_tag eq "source") {
                        $organism = $objFeatures->get_tag_values('organism');
                        }
                }

                for($seq->species->classification()){
                  $organism =  $organism . $_ . ";";
                }

                $organism   =~ s/["']/\_/g; # retirando aspas para inserir no bd

                # interação durante o processamento
                print "Locus : ", $locus, "\n";
                print "Definition : ", $definition, "\n";
                print "Gi: ", $gi, "\n";
                print "Keywords: ", $keywords, "\n";

                print "organism: ", $organism , "\n";
                print "Alphabet: ", $alphabet , "\n";

                $cont = $cont + 1;

                print $cont, "\n\n";

                &sequencias($locus, $definition, $gi, $keywords, $organism, $sequence, $arquivo,
                $alphabet);
  }

  $dbh->disconnect;
  close($fileName);


sub sequencias {

                my $query = "call pr_sequencias('" .  $_[0] . "','".  $_[1] . "','".  $_[2] . "','" .  $_[3] . "','"
                .  $_[4] . "','" .  $_[5] . "','" .  $_[6] . "','" .  $_[7] . "');";

                $dbh->do($query);

}
exit;
```

APÊNDICE C - Scripts escritos em PERL utilizados para inserir registros na tabela SIMILARIDADE

```
use DBI; # biblioteca perl de acesso a banco de dados
use Bio::SeqIO; # biblioteca perl de biotecnologia
use Bio::SearchIO;
use Bio::Tools::Run::StandAloneBlast;
use Bio::Search::HSP::HSPI

  $database = "xxx"; # nome do banco de dados
  $host = "xxx"; # nome da maquina
  $usuario = "xxx"; # usuario
  $senha = "xxx"; # senha

  # conectando ao banco de dados
  my $dbh = DBI->connect("DBI:mysql:database=$database;host=$host","$usuario",
"$senha",{'RaiseError' => 1});

  # definindo o programa da plataforma BLAST e o banco de dados
        @params = (program => 'blastp', database => 'NOME_ARQUIVO_FASTA.fasta'); #
        programa para comparacao entre sequencia proteicas
        $factory = Bio::Tools::Run::StandAloneBlast->new(@params);

  #print "\n Id: " , $ARGV[1], "\n\n";
  #my $id_seq_alvo = $ARGV[1];


        my $query = "select ID_SEQ, SEQ from SEQUENCIA WHERE length(SEQ) > 0 order by
        POSICAO_TOT"; # Todas sequências

        my $query = "select ID_SEQ, SEQ from SEQUENCIA WHERE COMPLETE_SEQ is null
        AND length(SEQ) > 0 order by POSICAO_PROP"; # Sequências parciais

        my $query = "select ID_SEQ, SEQ from SEQUENCIA WHERE COMPLETE_SEQ = '1'
        AND length(SEQ) > 0 order by POSICAO_PROC"; # Sequências completas

        my $sth = $dbh->prepare($query);
        my $res = $sth->execute();

        my $qtd = 0;
        my $qtdPorSequencia = 0;

        while(($id, $sequencia) = $sth->fetchrow_array) {

         # definindo a sequencia que sera comparada com todas do banco de dados
         $input = Bio::Seq->new(-id=>"Seq",-seq=>$sequencia);

         # executando o BLAST
         $blast_report = $factory->blastall($input);

         $qtdPorSequencia = 0;

         while (my $result = $blast_report->next_result)  {
                  while (my $hit = $result->next_hit)      {
                           while (my $hsp = $hit->next_hsp) {
```

```
                                # interatividade durante o processo
                                 print "Nome: " , $hit->name,
                                 " per: " , $hsp->percent_identity,
                                 " Score: " , $hsp->score,
                                 " frac_identical: " , $hsp->frac_identical(),
                                 " num_identical: " , $hsp->num_identical(),
                                 " E: ", $hsp->evalue,
                                 " length: ", $hsp->length,
                                 " gaps: ", $hsp->gaps,
                                 " frac_identical: ", $hsp->frac_identical,
                                 " frac_conserved: ", $hsp->frac_conserved,
                                 " n: ", $hsp->n,
                                 " matches: ", $hsp->matches,
                                 " num_identical: ", $hsp->num_identical,
                                 " num_conserved: ", $hsp->num_conserved,
                                 "\n\n";

                                  $string = $hsp->evalue;
                                  $string = &Replace($string, ',', '');

                                &inserir($id, $hit->name, $hsp->percent_identity, $hsp->score, $string,
                                $hsp->length, $hsp->gaps, $hsp->frac_identical, $hsp->frac_conserved,
                                $hsp->n, $hsp->matches, $hsp->num_identical, $hsp->num_conserved);

                                $qtdPorSequencia = $qtdPorSequencia + 1;
                          }
                        }
                }

                $qtd = $qtd + 1;

                print "Seq: ", $qtd, " n: ", $qtdPorSequencia, "\n";

                }

                $sth->finish();

                print "\nQtd: ", $qtd, "\n\n";

        # Removendo espaços em branco do inicio e final das strings
        sub Replace {

                my $strString = shift;
                my $strSearch = shift;
                my $strReplace = shift;
                $strString =~ s/$strSearch/$strReplace/ge;
                return $strString;
        }

        sub inserir {

                my $query = "call pr_similaridade (" .  $_[0] . ", ".  $_[1] . ",'".  $_[2] . "','".  $_[3] . "','" .  $_[4] .
                "','" .  $_[5] . "','" .  $_[6] . "','" .  $_[7] . "','" .  $_[8] . "','" .  $_[9] . "','" .  $_[10] . "','" .  $_[11] .
                "','" .  $_[12] . "');";
```

```
        my $sth = $dbh->prepare($query);
        my $res = $sth->execute();
}
exit;
```

APÊNDICE D - Scripts escritos em PERL utilizados para construção da matriz de similaridade

```
use Bio::SeqIO; # biblioteca perl de biotecnologia
use DBI; # biblioteca perl de acesso a banco de dados


        $database = "xxx"; # nome do banco de dados
        $host = "xxx"; # nome da maquina
        $usuario = "xxx"; # usuario
        $senha = "xxx"; # senha

        # conectando ao banco de dados
        my $dbh = DBI->connect("DBI:mysql:database=$database;host=$host","$usuario",
        "$senha",{'RaiseError' => 1});

        # Proteinas Todas
        my $query = "select DISTINCT ID_SEQ, POSICAO_TOT from SEQUENCIA,
        SIMILARIDADE where ALPHABET = 'Protein' AND ID_SEQ = COD_SEQ_ALVO order by
        POSICAO_TOT";

        # Proteinas parciais
        my $query = "select DISTINCT ID_SEQ, POSICAO_PROP from SEQUENCIA,
        SIMILARIDADE where COMPLETE_SEQ is null and ALPHABET = 'Protein' AND ID_SEQ
        = COD_SEQ_ALVO order by POSICAO_PROP";

        # Proteinas completas
        my $query = "select DISTINCT ID_SEQ, POSICAO_PROC from SEQUENCIA,
        SIMILARIDADE where COMPLETE_SEQ = '1' and ALPHABET = 'Protein' AND ID_SEQ =
        COD_SEQ_ALVO order by POSICAO_PROC";


        my $sth = $dbh->prepare($query);
        my $res = $sth->execute();

        my $qry = "select count(*) as result from SEQUENCIA WHERE length(SEQ) > 0"; # Todas

        my $qry = "select count(*) as result from SEQUENCIA WHERE COMPLETE_SEQ is null
        AND length(SEQ) > 0"; # parciais

        my $qry = "select count(*) as result from SEQUENCIA WHERE COMPLETE_SEQ = '1'
        AND length(SEQ) > 0"; # completas

        my $pro = $dbh->prepare($qry);
        my $p = $pro->execute();

        my $seqAlvo = 0;

        my $i = 1; # sequência que esta sendo trabalhada

        my $tot = $pro->fetchrow_array; # Pegando quantidade total de proteínas

        while(($id) = $sth->fetchrow_array) {

                $seqAlvo = $id;
```

```
print "Sequencia ->  $seqAlvo \n";

# Sequências completas
my $qry = "select distinct p.POSICAO_PROC, FLOOR(PERCENTUAL) as
PERCENTUAL from(select COD_SEQ_SIMILAR, PERCENTUAL from
SEQUENCIA p, SIMILARIDADE s where ID_SEQ = $seqAlvo and ID_SEQ =
COD_SEQ_ALVO and COD_SEQ_ALVO <> COD_SEQ_SIMILAR) as temp ,
SEQUENCIA p where p.ID_SEQ = COD_SEQ_SIMILAR order by
p.POSICAO_PROC;";

# Sequências parciais
        my $qry = "select distinct p.POSICAO_PROP, FLOOR(PERCENTUAL) as
PERCENTUAL from(select COD_SEQ_SIMILAR, PERCENTUAL from
SEQUENCIA p, SIMILARIDADE s where ID_SEQ = $seqAlvo and ID_SEQ =
COD_SEQ_ALVO and COD_SEQ_ALVO <> COD_SEQ_SIMILAR) as temp ,
SEQUENCIA p where p.ID_SEQ = COD_SEQ_SIMILAR order by
p.POSICAO_PROP;";

# Todas sequências
        my $qry = "select distinct p.POSICAO_TOT, FLOOR(PERCENTUAL) as
PERCENTUAL from(select COD_SEQ_SIMILAR, PERCENTUAL from
SEQUENCIA p, SIMILARIDADE s where ID_SEQ = $seqAlvo and ID_SEQ =
COD_SEQ_ALVO and COD_SEQ_ALVO <> COD_SEQ_SIMILAR) as temp ,
SEQUENCIA p where p.ID_SEQ = COD_SEQ_SIMILAR order by
p.POSICAO_TOT;";

my $simi = $dbh->prepare($qry);
my $sim  = $simi->execute();

while(($posicao, $similaridade) = $simi->fetchrow_array) {

   $resultBanco{$posicao} = {
     similaridade => $similaridade,
   };
}

my $temp = 0;

for( $k = 1 ; $k <= $tot ; $k++ ) {

    # montando matriz
    for $j (keys %resultBanco) {
      if($k == $j) {
           $temp = $resultBanco{$j}->{similaridade};
      }
  }

  $matriz{$i}{$k} = { n => $temp};
  $temp = 0;
}

print "  Gerou\n\n";

# limpando lista
for $u (keys %resultBanco) {
```

```
                delete $resultBanco{$u};
            }

            $i = $i + 1;

            $simi->finish();
        }

        $sth->finish();

        my $tempX = 0; # linha
        my $tempY = 0; # coluna

        print "Percorrendo da matriz \n\n";

        # percorrendo da matriz para forcar a similaridade
        for( $i = 1 ; $i <= $tot ; $i++ ) {

            for( $j = $i ; $j <= $tot ; $j++ ) {

                $tempX = $matriz{$i}{$j}->{n};
                $tempY = $matriz{$j}{$i}->{n};

                print " tempX: $tempX  tempY: $tempY \n";

                # forçando similaridade, atribuindo o valor do maior
                if ($tempX > $tempY) {
                    $matriz{$i}{$j} = { n => $tempX};
                    $matriz{$j}{$i} = { n => $tempX};
                } else {
                    $matriz{$i}{$j} = { n => $tempY};
                    $matriz{$j}{$i} = { n => $tempY};
                }
            }

            $tempX = 0;
            $tempY = 0;
        }

        print "Escrevendo matriz no arquivo \n\n";

        open(sequencias, ">NOME_ARQUIVO_RESULTADO.txt"); # Abrir arquivo para escrita

        for( $i = 1 ; $i <= $tot ; $i++ ) {

          for( $j = 1 ; $j <= $tot ; $j++ ) {
            print sequencias $matriz{$i}{$j}->{n}. " ";
          }
          print sequencias "\n";
        }

        close(sequencias);

        print "Finalizado \n\n";
exit;
```

APÊNDICE E – Programa escrito em C utilizado para construção das matrizes de adjacência

```
#include <stdio.h>
#include <string.h>

int main() {

	int tam;

	// tamanho do buffer
	tam = 5000;

	// variáveis de trabalho
	char buffer[tam], nome[10], nqtdArestas[3], temp[10];
	int  i, j, cont, n, qtdArestas;

	// definindo matriz com de trabalho
	n = 327; // n eh a quantidade de proteínas
	int matriz[n][n];

	// definição dos arquivos de entrada e saída
	FILE *inputfile;
	FILE *outputfile;

	// definindo arquivo de saída
	outputfile = fopen("NOME_ARQUIVO.TXT", "w");

	// a variável cont inicia com o menor limiar de similaridade no banco de dados
	cont = 17;

	// While que ler as matrizes de adjacência
	while (cont <= 100) {

		printf("\nLendo arquivo %i", cont);
		printf("\n");

		// definir nome do arquivo de entrada
		sprintf(nome, "%d", cont);

		strcpy(temp,nome);

		inputfile = fopen(strcat(nome,".txt"), "r");

		i = 0;
		// lendo o arquivo de entrada
		while (fgets(buffer, tam, inputfile)) {

		    for(j = 0 ; j < strlen(buffer) ; j++) {

			if ((buffer[j] != ' ' ) && (buffer[j] != '\n' )) {

				matriz[i][j] = buffer[j];
```

```
                    }
                }

                i++; // incrementando a linha da matriz[n][n]
            }

            fclose(inputfile);

            qtdArestas = 0;

            // contando quantidade de arestas
            for(i = 0 ; i < n ; i++) {
                for(j = i ; j < n ; j++) {
                    // se o elemento A(i,j) for igual a 1, incrementa 1 a variável de trabalho
                    if (matriz[i][j] == 49) {
                        qtdArestas++;
                    }
                }
            }

            sprintf(nqtdArestas, "%d", qtdArestas);

            // escrevendo uma linda no arquivo de saída
            fputs( temp, outputfile);
            fputs( ",", outputfile);
            fputs( nqtdArestas, outputfile);
            fputs( "\n", outputfile);

            cont = cont + 1;

        }// fim While que ler as matrizes de adjacência

        fclose(outputfile);

        printf("\n\nFinalizado \n\n");
}
```

APÊNDICE F – Programa escrito em Java utilizado para definição da distância euclidiana.

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.util.ArrayList;
import java.text.DecimalFormat;
import java.math.BigDecimal;

public class Distancia {

	private static Distancia instance = null;

	public static ArrayList<String> matrizX = null;

	public static ArrayList<String> matrizY = null;

	public static void main(String[] args) throws Exception {

		Distancia distancia = Distancia.getInstance();

		ArrayList<String> matrizX = distancia.gematrizX();

		ArrayList<String> matrizY = distancia.getmatrizY();

		DecimalFormat aproximador = new DecimalFormat( " 0.00 " );

		System.out.println("Inicio do processamento ... \n");

		int ini = 19;  // menor grau de similaridade entre as sequências

		int fim = 100; // maior grau de similaridade entre as sequências

		int arquivo1 = ini - 1;

		int arquivo2 = fim - 1;

		while (arquivo2 < fim) {

			arquivo1 = arquivo1 + 1;

			arquivo2 = arquivo1 + 1;

			distancia.lerMatriz("Inicio_nome_arquivo_"+ arquivo1 + ".txt", matrizX);

			distancia.lerMatriz("Inicio_nome_arquivo_"+ arquivo2 + ".txt", matrizY);

			System.out.println(arquivo1 + "  " +
			aproximador.format(distancia.distanciaEuclidiana()));

			distancia.limparMatrizes();

		}

		System.out.println("\nFim do processamento");
```

```
}

public static Distancia getInstance() {

	if (instance == null) {
		instance = new Distancia();
	}
	return (instance);
}

public ArrayList<String> gematrizX() {
	if (matrizX == null) {
		matrizX = new ArrayList<String>();
	}
	return (matrizX);
}

public ArrayList<String> getmatrizY() {
	if (matrizY == null) {
		matrizY = new ArrayList<String>();
	}
	return (matrizY);
}

public void lerMatriz(String nomeArquivo, ArrayList<String> arrayList) throws Exception {

	String linha = null;

	FileReader arquivo = new FileReader(nomeArquivo);

	BufferedReader leitor = new BufferedReader(arquivo);

	while((linha = leitor.readLine()) != null) {

		if (!linha.trim().startsWith("matriz")){

			for (int i = 0; i < linha.length(); i++) {

				if(linha.charAt(i) != ' '){
					arrayList.add(String.valueOf(linha.charAt(i)).trim());
				}
			}
		}
	}

	leitor.close();

	arquivo.close();
}

public double distanciaEuclidiana() throws Exception {

	Double D = new Double(0);

	double soma = 0;
```

```
                double tam = matrizX.size();

        for ( int i = 0; i < tam; i++ ) {

            try {
                soma = soma + Math.pow((D.valueOf(matrizX.get(i)).doubleValue() -
                D.valueOf(matrizY.get(i)).doubleValue()), 2);
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }

        }

        soma = Math.sqrt(soma);

        return soma;

            }

            public void limparMatrizes() {

                matrizX.clear();
                matrizY.clear();
            }
}
```

APÊNDICE G – Programa Java utilizado para definir o percentual de congruência das redes complexas e dos métodos tradicionais de filogenia.

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.util.ArrayList;

public class EdendoXFilo_Atualizado {


	private static EdendoXFilo_Atualizado instance = null;

	private ArrayList<Grupo> gruposDendograma = null;

	private ArrayList<Grupo> gruposFilogenia = null;

	public static void main(String[] args) throws Exception {

		EdendoXFilo_Atualizado edendoXArvore = EdendoXFilo_Atualizado.getInstance();

		ArrayList<Grupo> filogenia = edendoXArvore.getGruposFilogenia();

		ArrayList<Grupo> dendograma = edendoXArvore.getGruposDendograma();

		edendoXArvore.lerArquivo("nome_arquivo_dendograma.txt", dendograma);

		edendoXArvore.lerArquivo("nome_arquivo_filograma.txt", filogenia);

		edendoXArvore.compararGrupos_FESC(dendograma, filogenia);
	}

	public static EdendoXFilo_Atualizado getInstance() {

		if (instance == null) {
			instance = new EdendoXFilo_Atualizado();
		}
		return (instance);
	}

	public ArrayList<Grupo> getGruposFilogenia() {

		if (gruposFilogenia == null) {
			gruposFilogenia = new ArrayList<Grupo>();
		}
		return (gruposFilogenia);
	}

	public ArrayList<Grupo> getGruposDendograma() {

		if (gruposDendograma == null) {
			gruposDendograma = new ArrayList<Grupo>();
		}
		return (gruposDendograma);
	}
```

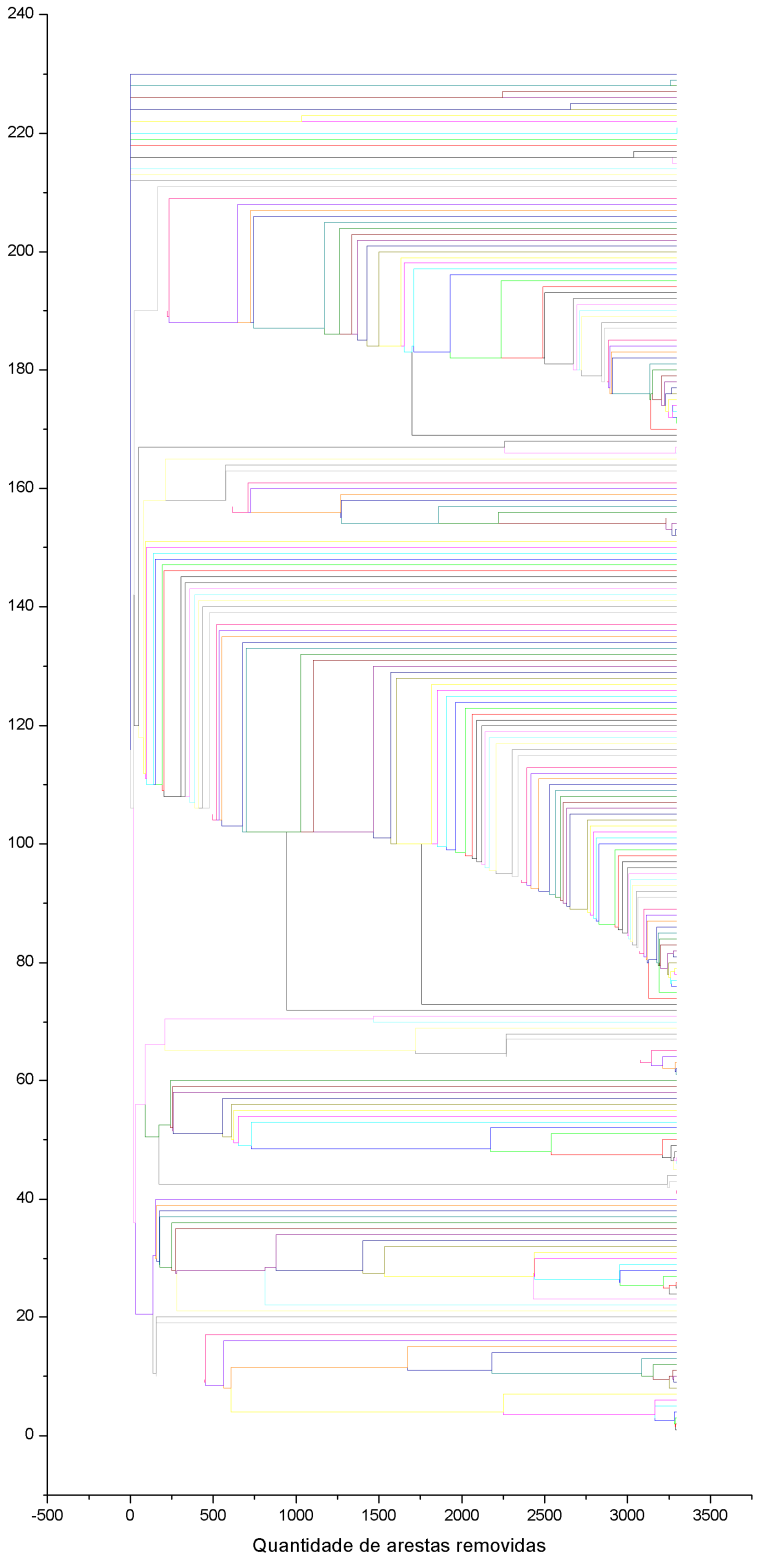
```
	public void lerArquivo(String nomeArquivo, ArrayList<Grupo> arrayList) throws Exception {

		String linha = null;

		FileReader arquivo = new FileReader(nomeArquivo);

		BufferedReader leitor = new BufferedReader(arquivo);

		Grupo tempGrupo;

		String temp;

		while((linha = leitor.readLine()) != null) {

			tempGrupo = new Grupo();

			temp = "";

			for (int i = 0; i < linha.length(); i++) {

				if(linha.charAt(i) != ' ') {
					temp = temp + linha.charAt(i);
				}else{
					tempGrupo.addLocus(temp);
					temp = "";
				}
			}

			arrayList.add(tempGrupo);
		}

		leitor.close();

		arquivo.close();
	}

	public void compararGrupos_FESC(ArrayList<Grupo> dendograma, ArrayList<Grupo>
filogenia){

		int [][] tabela;

		ArrayList<Integer> melhorCombinacao = new ArrayList<Integer>();

		ArrayList<Integer> posicaoDiferente = new ArrayList<Integer>();

		int total = obterTotalLocus(filogenia);

		int erro = 0;

		int acumulador = 0;

		int tamanho = 5;

		int menorLinha = 0;
```

```
		int menorPosicao = 0;
		boolean continuarLoop = true;
		double identidade = (double)0.0;
		double acum = (double)0.0;
		tabela = new int[tamanho][tamanho];

		for (int i = 0; i < filogenia.size(); i++) {
			erro = 0;
			acumulador = 0;
			for (int j = 0; j < dendograma.size(); j++) {
				erro = locusIguais(filogenia.get(i), dendograma.get(j));
				tabela[i][j] = erro;
				acumulador = acumulador + erro;
			}
		}

System.out.println("================================================\n");
		System.out.println("Matriz de incompatibilidade\n");
		for (int i = 0; i < tamanho; i++) {
			for (int j = 0; j < tamanho; j++) {
				System.out.print(tabela[i][j] + " ");
			}
			System.out.println("\n");
		}
		System.out.println("As melhores combinações\n");
		identidade = (double)0.0;
		acum = (double)0.0;
		posicaoDiferente.clear();
		for (int i = 0; i < tamanho; i++) {
			continuarLoop = true;
			while (continuarLoop){
```

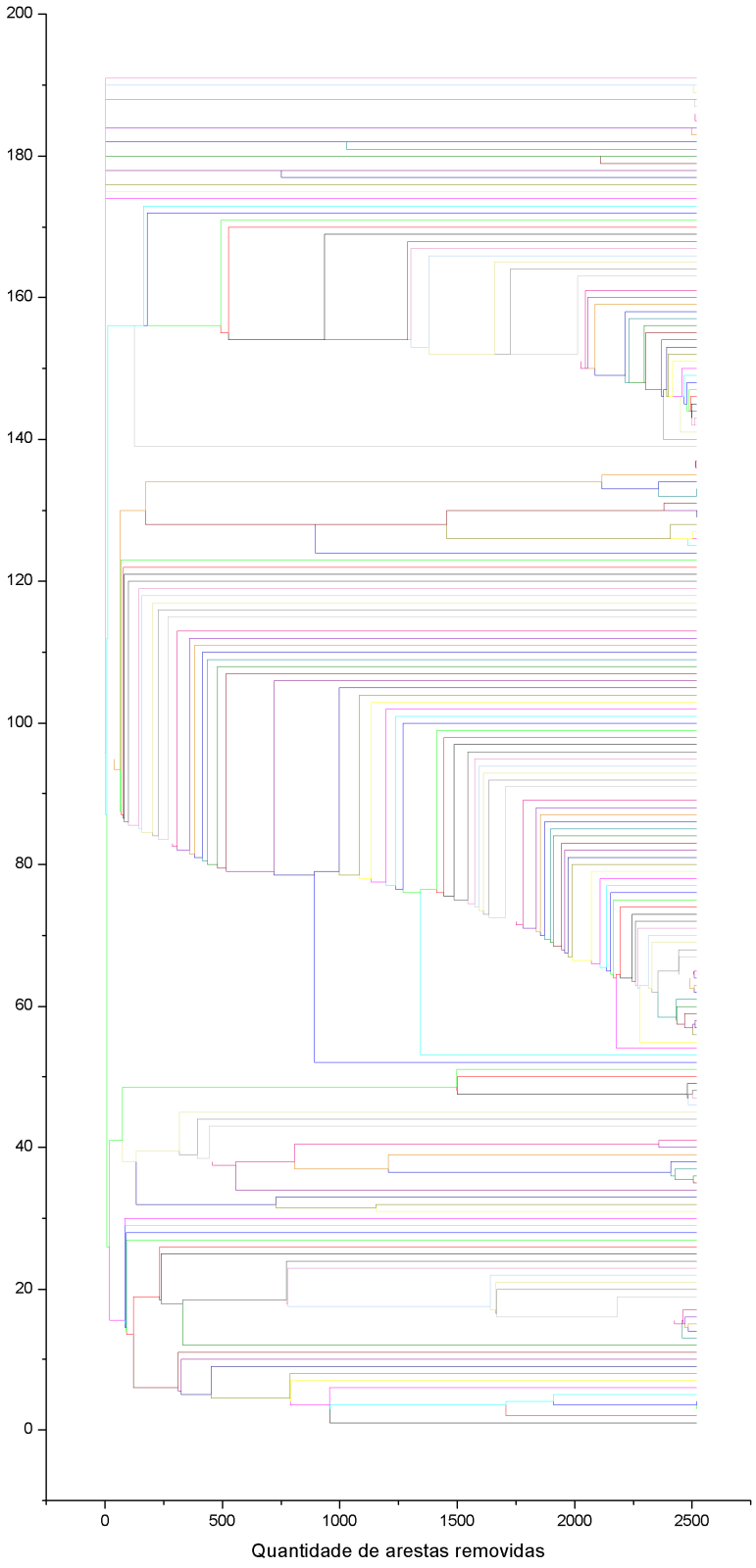
```
                                menorPosicao = obterMenorPosicaoDaLinha(i, tabela, tamanho,
posicaoDiferente);
                                if (!melhorCombinacao.contains(menorPosicao)){
                                        melhorCombinacao.add(menorPosicao);
                                        continuarLoop = false;
                                }else{
                                        posicaoDiferente.add(menorPosicao - 1);
                                        continuarLoop = true;
                                }
                        }
                        menorLinha = tabela[i][menorPosicao-1];
                        System.out.println("Grupo (" + (i + 1) + ")" + "(" + menorPosicao + ")");
                        identidade = (filogenia.get(i).getLocus().size() - menorLinha) / (double)total;
                        acum = acum + identidade;
                }
                System.out.println("\nIdentidade: " + (acum) + "\n");
        }
        public int locusIguais(Grupo filo, Grupo dendo){
                boolean eH;
                boolean ehIgual;
                int erro = 0;
                for (int i = 0; i < filo.getLocus().size(); i++) {
                        eH = false;
                        ehIgual = false;
                        for (int j = 0; j < dendo.getLocus().size(); j++) {
                                ehIgual =
(filo.getItemLocus(i).equalsIgnoreCase(dendo.getItemLocus(j)));
                                if (ehIgual) {
                                        eH = true;
                                }
                        }
                        if (!eH) {
```

```
                                        erro = erro + 1;
                                }
                        }

                        return(erro);
                }

                public int obterTotalLocus(ArrayList<Grupo> arrayList){

                        int total = 0;

                        for (Grupo grupo : arrayList) {

                                total = total + grupo.getLocus().size();
                        }

                        return (total);
                }

                public int obterMenorPosicaoDaLinha(int linha, int [][] tabela, int tamanho,
ArrayList<Integer> posicaoDiferente){

                  int res = 0;

                  int menorIndice = 0;

            for (int j = 0; j < tamanho; j++) {

                  if (! posicaoDiferente.isEmpty()){

                              if ((!posicaoDiferente.contains(j)) && (tabela[linha][j] <=
tabela[linha][menorIndice])){

                                  menorIndice = j;
                        }

                              if ((!posicaoDiferente.contains(j)) && (tabela[linha][j] > tabela[linha][menorIndice])
&& (posicaoDiferente.contains(menorIndice))){

                                  menorIndice = j;
                        }

                  }else{

                        if (tabela[linha][j] < tabela[linha][menorIndice]){

                              menorIndice = j;
                        }
                  }
            }

                  res = (menorIndice + 1);

                  return (res);

} }
```

APÊNDICE H – Dendogramas das sequências totais de proteína.



APÊNDICE I – Dendogramas das sequências parciais de proteína.



APÊNDICE J – Estatísticas do banco de dados.

Tabela 07 – Quantidade de sequências por organismo

Organismos	Quantidade
Agaricus	4
Amanita	16
Bensingtonia	2
Chondrostereum	1
Collybia	1
Coprinopsis	17
Corticium	1
Cryptococcus	33
Filobasidiella	1
Hypholoma	2
Laccaria	25
Lentinula	2
Malassezia	29
Moniliophthora	36
Phanerochaete	1
Phlebia	1
Pleurotus	6
Postia	4
Puccinia	6
Resinicium	1
Schizophyllum	3
Sporobolomyces	5
Stereum	2
Trametes	1
Tricholoma	2
Ustilago	28

Tabela 08 – Quantidade de sequências completas por organismo

Organismos	Quantidade
Agáricus	1
Coprinopsis	1
Cryptococcus	20
Lentinula	2
Malassezia	1
Moniliophthora	1
Pleurotus	6
Puccinia	5
Ustilago	2

Tabela 09 – Quantidade de sequências parciais por organismo

Organismos	Quantidade
Agaricus	3
Amanita	16
Bensingtonia	2
Chondrostereum	1
Collybia	1
Coprinopsis	16
Corticium	1
Cryptococcus	13
Filobasidiella	1
Hypholoma	2
Laccaria	25
Malassezia	28
Moniliophthora	35
Phanerochaete	1
Phlebia	1
Postia	4
Puccinia	1
Resinicium	1

Schizophyllum	3
Sporobolomyces	5
Stereum	2
Trametes	1
Tricholoma	2
Ustilago	26

APÊNDICE L – Artigo aceito para publicação, Biosystems, 2010.