Aspectos de genômica comparativa

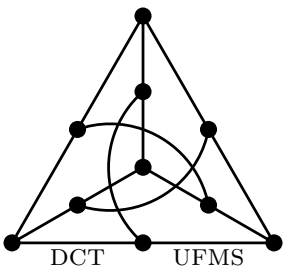
Carlos Juliano Moura Viana

Dissertação de Mestrado

Orientação:Prof.Dr. Nalvo Franco de Almeida Jr.

Área de Concentração: Biologia Computacional

Durante a elaboração desse trabalho o autor recebeu apoio financeiro da CAPES.



Departamento de Computação e Estatística
Centro de Ciências Exatas e Tecnologia
Universidade Federal de Mato Grosso do Sul
Agosto/2006

Aspectos de genômica comparativa

Este exemplar corresponde à redação final da tese devidamente corrigida e defendida por Carlos Juliano Moura Viana e aprovada pela comissão julgadora.

Campo Grande/MS, 31 de agosto de 2006.

Banca Examinadora:

- Prof. Dr. Nalvo Franco de Almeida Junior - Orientador (DCT-UFMS)
- Profa. Dra. Maria Emília Machado Telles Walter (CIC-UNB)
- Prof. Dr. Said Sadique Adi (DCT-UFMS)

à minha amada Amandita
e aos meus pais.

Agradecimentos

Tenho a oportunidade neste momento de fazer os meus agradecimentos a todas as pessoas que me ajudaram a transpor mais uma etapa na minha vida.

Gostaria primeiramente de agradecer a Deus pela saúde, pois sem ela nada disso seria possível.

Agradeço a minha amada família pelo carinho e pelo pleno apoio para continuar os meus estudos.

Um agradecimento em especial à minha amada *Amandita* pela sua enorme paciência, atenção, carinho, dedicação e pelos seus conselhos, que não me permitiram desanimar jamais diante dos momentos difíceis.

Ao professor Edson Norberto Cáceres que não mede esforços para melhorar os cursos de Bacharelado e Mestrado em Ciência da Computação nesta universidade. Ao professor Marcelo Henriques de Carvalho pelas suas dicas e colaboração com os trabalhos que desenvolvi no mestrado. Agradeço ao professor Henrique Mongelli pelas recomendações, que trouxeram-me novamente para o término desse trabalho.

Gostaria de agradecer em especial, ao meu orientador, professor Nalvo Franco de Almeida Junior, pela paciência, conselhos, dicas e atenção dedicados, desde o projeto final de curso até a conclusão deste trabalho de mestrado.

As minhas ex-colegas de sala, Graziela e Luciana, pelas valiosas discussões em bioinformática. A minha amiga Bianca (B1), pelos incentivos, auxílios e pelas divertidas conversas durante todo o mestrado. Aos meus amigos Cristiano, Anderson e Márcio pelos momentos descontraídos que aliviaram os momentos difíceis.

Em especial, ao meu amigo Cristiano, pelo convite e oportunidade de continuar trabalhando durante o final do mestrado, e pelos seus valiosos conselhos. Agradeço também ao pessoal do Laboratório de Engenharia de Software (LEDES) pelo ótimo acolhimento.

Enfim, agradeço desejando um muito obrigado a todos que me ajudaram de alguma forma a concluir esse trabalho.

Resumo

Com o avanço no seqüenciamento de genomas e facilidade no acesso às seqüências, técnicas computacionais de análise comparativa tornaram-se indispensáveis ferramentas para uma melhor caracterização e compreensão dos organismos em estudo. Um projeto genoma consiste, basicamente, em 3 grandes fases: seqüenciamento, anotação e análise. A segunda fase, anotação, consiste em determinar onde, em cada cromossomo, se encontram as regiões que codificam informações genéticas, os genes, assim como em determinar a caracterização funcional de cada gene. Na fase de análise, busca-se uma caracterização do organismo estudado, em termos de suas funcionalidades biológicas, a partir das informações geradas nas duas fases anteriores. Este trabalho está inserido no contexto da análise do genoma. Especificamente, o trabalho consiste na total reformulação do pacote de ferramentas denominado EGG. A reformulação inclui a reimplementação de todo o código-fonte, bem como a descrição e implementação de novas metodologias e funcionalidades. O objetivo principal é disponibilizar à comunidade científica um pacote com um conjunto de ferramentas para a comparação de genomas no nível dos seus genes e proteínas.

Abstract

With the advance in the genome sequencing and easiness in accessing the sequences, computational techniques of comparative analysis had become indispensable tools for a better characterization and understanding of the organisms in study. A genome project consists basically in 3 major phases: sequencing, annotation and analysis. The second phase, annotation, consists of determining, in each chromosome, where genetic coding regions are as well as in determining the functional characterization of each gene. In the analysis, the goal is a characterization of the organism of interest, in terms of its biological functionalities, from the information generated in the two previous phases. This work is inserted in the context of the analysis of the genome. Specifically, the work consists of the total reformularization of the package of tools named EGG. The reformularization includes reimplementing of the source code, as well as description and implementation of new methodologies and features. The main goal is to give to the scientific community a package with a set of tools for genome comparison in the level of its genes and proteins.

Sumário

Lista de Algoritmos

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Os avanços na área de Biotecnologia, em especial no desenvolvimento de técnicas de seqüenciamento de DNA, têm produzido uma gigantesca massa de dados biológicos, traduzidos em seqüências de DNA e de proteína. O grande desafio criado a partir da geração desses dados é a tarefa de analisá-los e transformá-los em informações biológicas relevantes, capazes de proporcionar aos pesquisadores novas habilidades. Essas habilidades incluem, por exemplo, novas técnicas para diagnóstico e tratamento de doenças genéticas. Em uma visão mais ampla, as descobertas feitas a partir dessas análises devem ser capazes de nos proporcionar um maior entendimento dos mecanismos biológicos que ditam as funcionalidades dos seres vivos.

A utilização da Ciência da Computação no tratamento dessas informações é imprescindível, não somente pela grande quantidade de dados gerados ou pelo tamanho das seqüências, mas também pela possibilidade de se desenvolverem novas técnicas computacionais para resolver problemas de Biologia Molecular. Essas novas técnicas deram origem a uma nova área de pesquisa, denominada Bioinformática ou Biologia Computacional.

O crescimento constante da Biologia Computacional tornou-se mais evidente com os inúmeros genomas seqüenciados nos últimos anos. Um projeto genoma consiste basicamente em três grandes fases: o seqüenciamento, anotação e análise. O seqüenciamento consiste na obtenção da seqüencia exata de nucleotídeos que compõe cada cromossomo do organismo estudado. A anotação é a tarefa de descobrir onde, em cada cromossomo, se encontram as regiões que codificam informações genéticas, denominadas genes, além da determinação da função biológica de cada um desses genes. A última fase, de análise, consiste em determinar as funcionalidades biológicas gerais do organismo, a partir das informações geradas na fase de anotação e seqüenciamento.

Este trabalho se insere na fase de análise de um genoma. Em particular, estamos interessados no desenvolvimento de técnicas e metodologias para a comparação de genomas. Essas comparações podem elucidar questões relacionadas a funcionalidades comuns e específicas importantes dos organismos. Uma forma de auxiliar na descoberta de tais informações relevantes passa, certamente, pela determinação dos papéis que os mais diversos objetos envolvidos num genoma desempenham. Esses papéis estão muitas vezes relacionados às características estruturais de cada objeto. Isso acontece de forma bem clara no caso de proteínas, que têm suas funções determinadas diretamente pela sua forma e estrutura.

Assim, é de se esperar que a comparação entre objetos, nas suas formas mais primárias, nos tragam pistas de relacionamentos entre eles e, por conseqüência, entre suas funcionalidades [1].

No caso de genomas é de se esperar, portanto, que a comparação entre seus objetos, especificamente seqüências de DNA e seus genes, seja útil na determinação de funcionalidades comuns. A idéia então é termos ferramentas que evidenciem aspectos funcionais comuns, além de proporcionarem uma melhor compreensão de como os genes se organizam nos diversos genomas.

Especificamente, a comparação de genomas tem como principais objetivos:

- detecção de similaridades e diferenças entre genomas completos, no nível de DNA;
- identificação de genes ou grupos de genes envolvidos em diversas funções;
- identificação de genes ou grupos de genes responsáveis por características fenotípicas peculiares a um genoma particular;
- identificação de genes homólogos (genes descendentes de um mesmo gene ancestral);
- anotação de genes de genomas não completos; e
- inferência de relações filogenéticas entre os organismos.

Histórico e Justificativa

Objetivando o desenvolvimento de ferramentas computacionais para a comparação de dois genomas, o trabalho desenvolvido por Almeida [1], em 2002, foi apresentado como tese de doutorado no Instituto de Computação da Unicamp. Uma das ferramentas propostas em [1], denominada EGG, compara dois proteomas. O proteoma de um organismo é o conjunto de proteínas expressas por seus genes. EGG foi inicialmente proposto em [3], reformulado em [4] e em [1], e sendo utilizado com sucesso em vários projetos genoma [9, 12, 15, 20, 21, 24, 26, 37, 45, 46]. EGG leva em consideração as posições relativas dos genes nos cromossomos.

Apesar das reformulações sofridas e apesar de ter sido intensamente utilizada, a ferramenta EGG ainda necessita de reformulações visando a melhoria de desempenho e portabilidade, além do acréscimo de novas funcionalidades. Este trabalho consiste na reformulação e reimplementação de EGG.

Sumário de resultados

Os principais resultados deste trabalho são:

- total reformulação do código dos programas contidos na ferramenta EGG;

- implementação de metodologia para encontrar regiões específicas – essa metodologia foi proposta em [1] mas não havia sido implementada;
- desenvolvimento e implementação de nova metodologia para encontrar genes parálogos (genes homólogos pertencentes ao mesmo organismo);
- desenvolvimento e implementação de uma versão de EGG, chamada EGG-LITE, para a comparação de genomas incompletos, onde se tem apenas genes, ou transcritos do organismo, sem o seqüenciamento completo dos cromossomos;
- desenvolvimento e implementação de módulo para a comparação de três proteomas.

Após essa reformulação, os códigos-fonte de EGG, além de manuais e documentação encontram-se disponíveis para *download* em `http://egg.dct.ufms.br/egg`.

A aplicação das novas metodologias inseridas em EGG foram utilizadas em trabalhos de pesquisa e resultaram na co-autoria das publicações [5, 41, 47]. Além disso, a experiência adquirida durante o programa de mestrado permitiu ainda a co-autoria de [44].

Organização do texto

O texto está organizado da seguinte forma. No Capítulo 2 alguns conceitos básicos e notações preliminares são apresentados. O Capítulo 3 traz a metodologia para a comparação de dois genomas. No Capítulo 4 a nova metodologia para a determinação de genes parálogos é apresentada. No Capítulo 5 a metodologia para a comparação de três genomas é descrita. Finalmente, o Capítulo 6 traz comentários conclusivos e propostas para trabalhos futuros.

Capítulo 2

Preliminares

Neste capítulo abordamos descrições de conceitos e notações utilizados no trabalho. Descrevemos alguns conceitos básicos de Biologia Molecular e de Computação. Estes conceitos permitem uma melhor interpretação e compreensão das metodologias descritas no capítulos posteriores.

Na Seção 2.1 descrevemos alguns fundamentos básicos da Biologia Molecular. A Seção 2.2 traz alguns fundamentos básicos de Computação. Na Seção 2.3 descrevemos conceitos envolvendo a comparação entre seqüências, como: *alinhamento*; o método de programação dinâmica para a comparação entre duas seqüências; e um método prático para a busca em bases de dados. Por fim, na Seção 2.4, descrevemos uma conhecida ferramenta utilizada na comparação de seqüências.

2.1 Fundamentos básicos de Biologia Molecular

Neste trabalho utilizamos **DNA**,(da língua inglesa ***D**eoxyribo**N**ucleic **A**cid*) para referenciar uma seqüência de letras escritas no alfabeto formado por `A`, `C`, `G`, `T`, e **RNA** (da língua inglesa **R**ibo**N**ucleic **A**cid) para referenciar uma seqüência de letras escritas no alfabeto `A`, `C`, `G`, `U`. Essas letras representam as bases nitrogenadas: **A**denina, **C**itosina, **G**uanina e **T**imina.

A **base nitrogenada** é um componente de uma estrutura básica denominada de nucleotídeo. Um **nucleotídeo** é um composto químico que consiste de uma molécula de açúcar denominada $2'$-*deoxyribose* no caso de DNA e $2'$-*ribose* no caso de RNA, por uma molécula de *fosfato* e por uma base nitrogenada. Existem quatro tipos diferentes de nucleotídeos, um para cada base, observando-se que a base nitrogenada `T` no DNA corresponde a base nitrogenada `U` no RNA.

A molécula de açúcar contém cinco átomos de carbono que são rotulados de $1'$ para $5'$. Os nucleotídos conectam-se através da ligação entre o carbono $5'$ de um nucleotídeo com o carbono $3'$ do nucleotídeo seguinte, utilizando a molécula de *fosfato*. Dessa forma, a mólecula resultante, denominada **cadeia simples de DNA**, possui por convenção, uma orientação, de $5'$ para $3'$.

A união dos nucleotídeos em duas cadeias (**fitas**) interligadas e anti-paralelas com a conformação de dupla-hélice formam a **seqüência de DNA**. Para designar um conjunto de possíveis nucleotídeos em uma determinada posição da seqüência de DNA, são admitidas letras adicionais. Esse alfabeto é descrito em [27]

O DNA inteiro de um organismo é denominado **genoma**. O genoma costuma variar em tamanho de acordo com a espécie, desde milhões de letras, no caso das bactérias, até bilhões de letras, no caso de mamíferos. Genomas são compostos também por longas seqüências de DNA, que costumam ser divididas em unidades denominadas de **cromossomos**. Um cromossomo é formado por duas cadeias (fitas) de DNA que se "torcem" uma sobre a outra.

As duas fitas de DNA são unidas pela ligação das bases de seus nucleotídeos. A base `A` sempre liga-se a base `T` e a base `G` sempre liga-se a base `C`. Os pares `A-T` e `C-G` são denominados pares de **bases complementares**. Esses pares são conhecidos como pares de base *Watson-Crick*. As duas fitas são ditas **anti-paralelas**, pois possuem **orientações**, da extremidade $5' \rightarrow 3'$, opostas uma em relação a outra.

No decorrer do texto, consideramos o DNA como uma seqüência de letras, onde cada letra representa uma base. Na Figura 2.1 apresentamos um exemplo da representação do DNA como duas seqüências de letras, com cada letra de uma seqüência justaposta a outra. As seqüências (fitas) de DNA são escritas uma sobre a outra revelando o pareamento entre as bases.

Uma fita é dita ser o **complemento-reverso** da outra. O complemento-reverso de um trecho de DNA G será denotado aqui por G^{CR}.

```
5'      ...  A T G G G C A C C G T G C G C  ...      3'
             | | | | | | | | | | | | | | |
3'      ...  T A C C C G T G G C A C G C G  ...      5'
```

Figura 2.1: Exemplo da representação de como as duas fitas de DNA pareiam.

Utiliza-se como unidade de medida de comprimento de um trecho de DNA, de fita dupla, o número de **pares de bases**, denotado por **bp** (do inglês *base pair*). Números maiores em geral são representadas por Kb (10^3 bp) e Mb (10^6 bp).

A seqüência de pares de bases do DNA de um organismo contém a informação necessária para a síntese de **proteínas**. As proteínas são seqüências de letras pertencentes ao alfabeto de 20 aminoácidos. Três nucleotídeos codificam um aminoácido. A tabela que permite corresponder cada tripla de nucleotídeos em um aminoácido é denominada **código genético**.

Combinando os 4 nucleotídeos em triplas, obtemos 64 combinações de possíveis triplas de nucleotídeos. Cada uma das combinações é denominada de **codon**. Como temos apenas 20 aminoácidos, mas 64 codons possíveis, temos que alguns codificam o mesmo aminoácido.

Dentre os 64 codons possíveis, 3 não especificam aminoácidos. Esses codons são denominados de codons de parada (ou **stop codons**), que sinalizam a terminação da tradução de uma seqüência do alfabeto de nucleotídeo para o alfabeto de aminoácidos. O código genético estabelece também um codon de início (ou **start codon**), que indica o início do processo de tradução.

Os trechos da seqüência de bases do DNA que são codificados em informação genética são denominados de **genes**. Existem dois tipos de genes, aqueles que codificam proteínas (a maioria) e aqueles que não codificam proteínas (codificam RNAs).

O conjunto de genes de um genoma que codificam proteínas é denominado de **proteoma**. Utilizamos os termos *gene* e *proteína* indistintamente, apesar de sabermos que um gene pode codificar mais de uma proteína e que existem genes que não codificam proteínas.

Na anotação de genomas tentamos predizer quais genes codificam uma ou mais proteínas. Dessa forma, comumente utilizamos o termo **proteína predita** de um determinado gene.

Em nosso trabalho, estamos interessados nos genes que são **homólogos**, ou seja, genes que evoluiram a partir de um gene ancestral comum. Especificamente, estamos interessados nos genes **ortólogos** e **parálogos**. Dois genes g e h são denominados de **ortólogos** se ambos descendem de um mesmo gene ancestral e pertencem a espécies distintas. Quando os genes g e h descendem de um mesmo gene ancestral, porém pertencem a um mesma espécie, g e h são denominados **parálogos**. A relação (g, h) entre genomas distintos é donominada "par de genes ortólogos".

Na seção seguinte abordamos alguns conceitos básicos computacionais com o objetivo de prover informações suficientes para a compreensão das metodologias descritas no decorrer do texto.

2.2 Fundamentos básicos de Computação

Uma **cadeia** é um sucessão de caracteres ou símbolos de um conjunto finito denominado de **alfabeto**. Utilizamos o termo **seqüência** como um sinônimo para o termo cadeia.

As seqüências podem ter símbolos repetidos, por exemplo $s =$ `TGCATT`. O **tamanho** de um seqüência s, denotado por $|s|$, é o número de símbolos em s. Para o exemplo anterior, $|s| = 6$. Um símbolo que ocupa a posição i em uma seqüência s é denotado por s_i. Logo, uma seqüência s é composta pelos símbolos $s_1, \ldots s_{|s|}$. Quando $|s| = 0$, denominamos s de **seqüência vazia**.

Apesar dos termos cadeia e seqüência possuírem o mesmo significado, os termos subcadeia e subseqüência representam conceitos distintos. Uma **subseqüência** de s é uma seqüência que pode ser obtida a partir de s pela remoção de alguns de seus símbolos. Considerando a seqüência exemplo s anteriormente, `GAT` é uma subseqüência de s, mas `GTAT` não é subseqüência de s. Uma **subcadeia** de s é uma cadeia formada pelos símbolos consecutivos de s, na mesma ordem em que aparecem em s. Considerando novamente a cadeia s como exemplo, `GCA` é uma subcadeia de s, mas `GAT` não é uma subcadeia de s.

Dado duas seqüências X e Y, dizemos que uma seqüência Z é uma **subseqüência comum** de X e Y se Z é uma subseqüência de ambos X e Y. Como exemplo, considere as

seqüências $A =$ `ACGTACAG` e $B =$ `TGAACC`. A seqüência `GAC` é uma subseqüência comum de A e B. Um **segmento** de uma seqüência é uma subcadeia da seqüência.

Alguns problemas computacionais envolvem subcadeias e subseqüências, como o problema de determinar uma subcadeia de máxima soma de uma cadeia de números reais S; sua generalização, que consiste em determinar não apenas uma, mas todas as subcadeias de máxima soma; e o problema de determinar a subseqüência comum mais longa (**LCS**). Podemos descrever o problema das subcadeias maximais e do LCS da seguinte forma:

Definição 2.1 (Problema das subcadeias maximais) *A entrada é uma cadeia ou seqüência $(x_1, x_2, \ldots, x_n)$, de números reais (não necessariamente positivos), denominada de "pontuação". O objetivo consiste em identificar todas as subcadeias que possuem maior pontuação, onde a pontuação $S_{i,j}$ de uma subcadeia $(x_i, x_{i+1}, \ldots, x_j)$ é obtida simplesmente pela soma de seus elementos:*

$$S_{i,j} = \sum_{k=i}^{j} x_k$$

Definição 2.2 (Problema da subseqüência comum mais longa - LCS) *Dadas duas seqüências $X = (x_1, x_2, \ldots, x_m)$ e $Y = (y_1, y_2, \ldots, y_n)$, desejamos encontrar a subseqüência comum de tamanho máximo entre X e Y.*

Uma melhor caracterização do problema do LCS pode ser encontrada em [14].

Um **grafo** G é uma tripla ordenada $(V(G), E(G), \psi_G)$ consistindo de um conjunto não vazio $V(G)$ de **vértices**, um conjunto $E(G)$ (disjunto de $V(G)$) de **arestas**, e uma função de incidência ψ_G que associa a cada aresta de G um par não ordenado de (e não necessariamente distinto) vértices de G. Se e é uma aresta e u e v são vértices tais que $\psi_G(e) = (u, v)$, então dizemos que e **liga** u a v. Os vértices u e v são denominados **extremos** de e. Na Figura 2.2 apresentamos um grafo com um conjunto de vértices $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ e com um conjunto de arestas $E = \{e_1, e_2, e_3, e_4, e_5\}$.

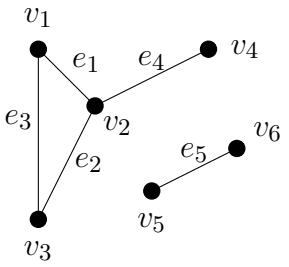


Figura 2.2: Exemplo de um grafo com seis vértices e cinco arestas.

Em um grafo G, se todo vértice é "atingível" pelos outros vértices, dizemos que o grafo G é **conexo**. Quando um grafo não é conexo, podemos determinar suas **componentes conexas**. A Figura 2.2 ilustra um grafo com duas componentes conexas.

Um grafo G é **bipartido** se o seu conjunto de vertices V pode ser particionado em dois conjuntos X e Y tais que, qualquer aresta (u, v) é tal que: $u \in X$ e $v \in Y$, ou $u \in Y$ e $v \in X$. Na Figura 2.3 apresentamos um grafo bipartido com cinco vértices e quatro arestas.

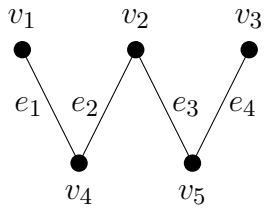


Figura 2.3: Exemplo de um grafo bipartido, tal que $X = \{v_1, v_2, v_3\}$ e $Y = \{v_3, v_4\}$.

Um conjunto C de vértices de um grafo $G(V, E)$ é uma **clique**, se para todo par u, v de vértices distintos em C, existe uma aresta $(u, v) \in E$. Uma clique C em G é **maximal**, se não existe outra clique C' em G que contenha C propriamente. As Figuras 2.4 e 2.5 ilustram, respectivamente, um exemplo de uma clique e de clique maximal no grafo G.

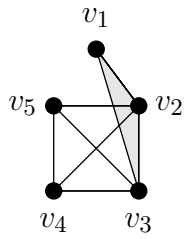


Figura 2.4: Exemplo de uma clique de tamanho 3 formada pelos vértices v_1, v_2 e v_3.

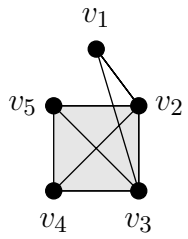


Figura 2.5: Exemplo de uma clique de tamanho 4 maximal formada pelos vértices v_2, v_3, v_4 e v_5.

Um Modelo Oculto de Markov (HMM) é uma base formal para construção de modelos probabilísticos. O modelo provê um conjunto de ferramentas conceituais para a construção de modelos complexos simplesmente pelo desenho de uma figura intuitiva [18]. Os modelos de Markov são bem adequados para muitas tarefas em Biologia Molecular e são o núcleo de uma diversa faixa de programas, incluindo os programas para procurar genes, buscar

por perfis, obter alinhamento múltiplo e identificar regiões regulatórias em DNA. Segundo Eddy [18], HMMs são os "legos" da análise computacional de seqüências.

Em uma descrição mais formal, um modelo oculto de markov $\mathcal{M}$ é definido por um alfabeto Σ, um conjunto de estados (escondidos) Q, uma matriz das probabilidades de transição de estados A, e uma matriz de probabilidades de emissão de símbolos E, mais especificamente:

- Σ é um alfabeto de símbolos;
- Q é um conjunto de estados que emitem símbolos do alfabeto Σ;
- $A = (a_{kl})$ é uma matriz $|Q| \times |Q|$ das probabilidades de transição de estados; e
- $E = (e_k(b))$ é uma matriz $|Q| \times |\Sigma|$ das probabilidades de emissão de símbolos;

Um caminho $\pi = \pi_1 \dots \pi_n$ em um HMM $\mathcal{M}$ é uma seqüência de estados. A probabilidade de que uma seqüência x de símbolos tenha sido gerada por uma seqüência de estados π dado um modelo $\mathcal{M}$ é :

$$P(x|\pi) = \prod_{i=1}^{n} P(x_i|\pi_i)P(\pi_i|\pi_{i+1}) = a_{\pi_0,\pi_1} \cdot \prod_{i=1}^{n} e_{\pi_i}(x_i) \cdot a_{\pi_i,\pi_{i+1}}$$

Convenientemente inserimos π_0 e π_{n+1} como os fictícios estados iniciais e terminais *início* e *fim*.

Na grande maioria das aplicações, deseja-se conhecer, dada uma seqüência x de símbolos, que é a parte não-oculta do modelo, qual a seqüência de estados (o caminho π) mais provável de ocorrer. Ou seja, π é o componente oculto do modelo.

Um simples exemplo é o de lançamento de 3 moedas, m_1, m_2, m_3, divulgando apenas a seqüência de resultados (cara ou coroa) e perguntando ao modelo qual é a seqüência de moedas lançadas, ou seja, para cada lançamento, qual foi a moeda lançada.

Neste caso,

- $\Sigma = \{cara, coroa\}$;
- $Q = \{m_1, m_2, m_3\}$;
- A é a matriz que indica a probabilidade do lançador trocar de moeda; e
- E é a matriz que dita o vício de cada moeda.

A utilização mais popular de HMM em Biologia Molecular é como um "perfil probabilístico" de uma família de proteínas, o qual é denominado de **perfil HMM** (pHMM). A partir de famílias de proteínas ou de DNA, um perfil HMM pode ser construído para buscar, em uma base de dados, por outros membros da família.

Um Perfil é uma simples representação de uma família de proteínas relacionadas, que é dada por uma alinhamento múltiplo. Dado um alinhamento múltiplo de n colunas em

um alfabeto $\mathcal{A}$, um **perfil** $\mathcal{P}$ é uma matriz $|\mathcal{A}| \times n$ que especifica a freqüência $e_i(a)$ de cada símbolo a de um alfabeto $\mathcal{A}$ em uma coluna i [28]. Mais adiante descreveremos o conceito de alinhamento.

Na literatura temos alguns textos que abordam HMMs e pHMMs, permitindo uma melhor compreensão dos conceitos aqui descritos. Especificamente, citamos o trabalho de Eddy [17], que descreve uma revisão sobre pHMMs; Salzberg e outros [33] fazem uma descrição não-matemática, no entanto mais didática, de HMMs, descrevendo-os através de um exemplo e posteriormente descrevendo perfis HMMs; Pevzner [28] descreve HMMs e pHMMs mais formalmente; Eddy [18] descreve em síntese o que é um odelo oculto de Markov; e um exemplo de HMM para encontrar promotores em uma seqüência de DNA procariótico é descrito em Almeida [2].

2.3 Comparação de seqüências

Comparação de seqüências é a mais importante operação em bioinformática, servindo como base para muitas outras manipulações mais complexas.

Ao compararmos duas seqüências, podemos determinar a similaridade e o alinhamento entre elas. A **similaridade** de duas seqüências é uma medida que indica o quanto as seqüências são semelhantes. O **alinhamento** de duas seqüências é uma forma de posicionar uma seqüência sobre a outra, acrescentando espaços nas duas seqüências, para que ambas fiquem do mesmo tamanho, com o objetivo de evidenciar a correspondência entre símbolos ou subseqüências similares das seqüências, ou seja, o alinhamento evidencia quão similares duas seqüências são.

Para computar a similaridade entre duas seqüências, poderíamos gerar todos os possíveis alinhamentos e escolher o melhor, ou seja, o de maior valor. Entretanto, o número de alinhamentos entre duas seqüências é exponencial. Uma solução eficiente para o problema é fornecida pela técnica de programação dinâmica.

A técnica de programação dinâmica consiste basicamente em solucinar uma instância do problema tirando vantagem de soluções anteriormente computadas de instâncias menores do mesmo problema. Na prática, a programação dinâmica soluciona todos os subproblemas somente uma vez, armazenando as soluções em uma tabela, para que sejam recuperadas sem a necessidade de uma recomputação. Uma excelente descrição da técnica, incluindo aplicações, pode ser encontradas no livro de Cormen [14]. Setubal e Meidanis [36] fazem uma descrição mais detalhada da comparação de duas seqüências, incluindo uma excelente descrição de algoritmos que utilizam a técnica de programação dinâmica para a comparação de seqüências biológicas.

Com o crescente volume de seqüências sendo geradas pelos laboratórios, bancos de dados de seqüências foram sendo criados. Esse evento criou uma necessidade por programas eficientes para comparar as seqüências nesses bancos de informações biológicas. Em essência, o problema consiste em listar quais trechos de uma seqüência de proteína ou de DNA são similares a quais regiões de uma determinada seqüência de proteína ou DNA fornecida pelo usuário.

Devido a complexidade quadrática dos algoritmos baseados em programação dinâmica, novos e rápidos métodos têm sido desenvolvidos, em especial heurísticas têm sido empregadas na prática. Um dos programas baseado em heurísticas, mais freqüentemente utilizado de busca de seqüências similares em bases de dados, é o BLAST, que será descrito a seguir.

2.4 BLAST

O BLAST, Basic Local Alignment Search Tool, foi inicialmente proposto por Altschul e colegas [6], e desde então vem sendo aperfeiçoado [7].

Dada uma seqüência de entrada, denominada de seqüência *query*, BLAST retorna uma lista de possíveis seqüências similares, denominadas *hits*, e seus alinhamentos com a seqüência *query*. Cada *hit* é acompanhado de uma estimativa de **significância estatística**, denominada *e-value*. Em essência, o **e-value** representa a quantidade de hits com uma determinada pontuação que são esperados ao acaso. Dessa forma, quando menor é o valor de *e-value*, menor é a probabilidade de um determinado *hit* ter sido encontrado ao acaso. Os alinhamentos entre a seqüência *query* e seus *hits* possuem um valor, denominado de *score*. O valor de *score* considera o tamanho do banco de seqüências e os tamanhos das seqüências.

A metodologia de BLAST para computar os *hits* é basicamente a seguinte. Primeiro, BLAST encontra certas "sementes", que são subcadeias curtas de tamanho W da seqüência *query*, cujo alinhamento com subcadeias de mesmo tamanho das seqüências da base de seqüências, tenha valor maior ou igual a uma valor limite T, utilizando para isso alguma matriz de substituição de aminoácidos. As "sementes" da seqüência *query* e da base de seqüências são então estendidas em ambas as direções, até o *score* máximo possível, para que a extensão dessa "semente" em particular seja alcançada. BLAST tem um critério de parar as extensões quando o valor de *score* cair a um valor de limite inferior X. Os segmentos estendidos são utilizados na construção do alinhamento.

Os valores de W, T e X são parâmetros de BLAST. Obviamente, todos os parâmetros de BLAST influenciam diretamente no resultado da busca por seqüências, fazendo com que a complexidade do BLAST seja difícil de calcular, visto que todos os parâmetros tornam-se fatores importantes na disputa entre sensibilidade e seletividade.

Capítulo 3

Comparação de dois genomas

Neste capítulo abordaremos a estratégia descrita em Almeida [1] para a comparação de dois genomas, que nos permita determinar regiões comuns em termos de genes conservados, determinar os genes específicos e as regiões específicas entre os proteomas, determinar os pares de genes ortólogos e as regiões ortólogas, construir o alinhamento entre os proteomas, e por fim comparar genomas não seqüenciados completamente. Objetivamos desenvolver uma ferramenta que ajude a explicar como a reordenação e o reagrupamento de genes influenciam nas diferenças entre as funcionalidades de dois genomas.

Estamos interessados nos genes que codificam proteínas, presentes nas fitas de DNA de uma espécie, onde cada proteína possui uma posição de acordo com a ordem em que o seu respectivo gene aparece no genoma. Assim, técnicas de comparação para as proteínas preditas dos genomas são necessárias.

Este capítulo está organizado da seguinte forma. Na Seção 3.1 descrevemos a metodologia utilizada. Na Seção 3.2 descrevemos uma proposta de implementação segundo a metodologia descrita. Na Seção 3.3 descrevemos as novas funcionalidades acrescentadas à nova proposta de implementação. Por fim, na Seção 3.4, descrevemos sucintamente uma comparação com outras ferramentas.

3.1 Metodologia

Descreveremos nessa seção a metodologia utilizada por Almeida em [1] que possibilita a comparação entre dois proteomas. Existem outras metodologias [22, 23, 31, 32, 39] que também permitem determinar regiões de elementos conservados entre as espécies comparadas. Na Seção 3.4 descreveremos algumas dessas outras metodologias.

Embora algumas definições tenham sido descritas anteriormente, no Capítulo 2, descreveremos e refaremos outras definições, para uma melhor interpretação das próximas seções.

- A **fita** de um gene é a fita de DNA a qual o gene pertence, sendo denominada de fita '+' ou '-';

- A **ordem dos genes** de um proteoma é dada pela ordem não-decrescente das coordenadas de início dos genes. Seja P_i a posição da primeira base de um gene g_i, caso g_i tenha sido codificado na fita '+'; ou a posição da última base antes do códon de terminação, caso g_i tenha sido codificado na fita '-'. Então a ordem dos genes, $g_1, g_2, \ldots, g_n$, é determinada respeitando a relação $P_1 \leq P_2 \leq \ldots \leq P_n$;

  Na Figura 3.1 temos duas representações gráficas simplificadas do proteoma de um genoma G. A primeira representação ilustra os genes e suas orientações: setas para a esquerda representam genes pertencentes a fita '-', enquanto que setas para a direita representam genes pertencentes a fita '+'. A segunda representação é mais adequada para nosso propósito, pois considera a ordem dos genes baseada nas orientações descritas acima.

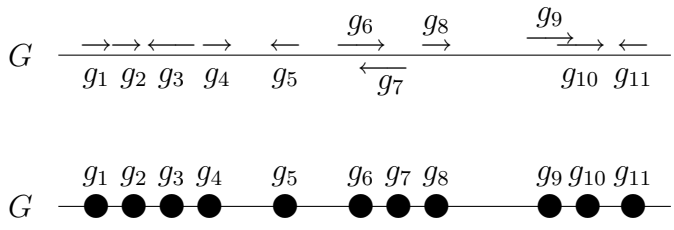


Figura 3.1: Exemplo de representação dos genes de um proteoma G.

- Dois genes g_i e g_j são **homólogos** se são descendentes de um mesmo gene ancestral;
- Dois genes g_i e g_j de um mesmo genoma G são **parálogos** se são homólogos e essa homologia originou-se através de um evento de duplicação;
- Uma **região de genes consecutivos** (**RGC**) é um conjunto de genes consecutivos em um proteoma, de acordo com suas coordenadas de início, independente da fita. Assim, temos que o próprio proteoma é uma RGC;
- Dois genes g de G e h de H são **ortólogos** se são homólogos em genomas diferentes através de um evento de especiação ocorrido antes de um evento de duplicação. Dizemos que (g, h) é um **par de ortólogos**;
- Um gene g de um proteoma G é **específico** em relação a um proteoma H se não existir gene h no proteoma H tal que g e h são ortólogos;
- Uma **região específica** (**RE**) de um proteoma G em relação a um outro proteoma H é uma região de G, denotada por $RE(G)$, tal que $|RE(G)| \geq E'$, onde E' é um limite fixo;
- Uma **região ortóloga** (**RO**) de dois proteomas G e H é um par (α, β) tal que:
  - α é uma RGC em G;
  - β é uma RGC em H;
  - α e β são descendentes de uma mesma região ancestral; e

– α e β contêm aproximadamente o mesmo número de genes.

Uma descrição mais formal e detalhada de região ortóloga será apresentada na Seção 3.1.3.

- Dois pares de genes ortólogos (g, h) e $(g^{'}, h^{'})$ formam um **cruzamento** quando a ordem de g e $g^{'}$ no proteoma G e a ordem de h e $h^{'}$ no proteoma H são invertidas; e

- A **espinha dorsal de duas RGCs**, α de G e β de H, é uma seqüência de pares de ortólogos (g, h), tal que:

  – cada gene em α tem no máximo um gene ortólogo a ele em β, e vice-versa; e

  – não existem cruzamentos entre os pares da seqüência.

As figuras abaixo ilustram, respectivamente, um exemplo de RO e de uma espinha dorsal de duas RGCs.

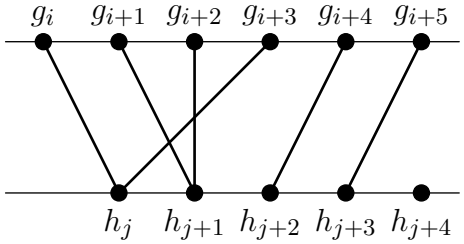


Figura 3.2: Representação de uma região ortóloga.

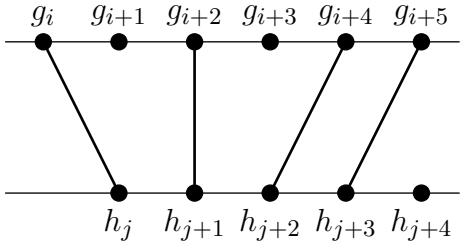


Figura 3.3: Representação de uma espinha dorsal de duas Regiões de Genes Consecutivos - RGCs.

Antes de apresentarmos a metodologia descrita por Almeida em [1], citaremos os objetivos que desejamos alcançar na comparação de dois proteomas:

1. Encontrar genes específicos entre proteomas;

2. Encontrar regiões específicas entre proteomas;

3. Encontrar pares de genes ortólogos;

4. Encontrar regiões ortólogas;

5. Determinar a espinha dorsal entre proteomas; e

6. Determinar famílias de genes parálogos de um proteoma.

Apresentaremos separadamente o item de número 6 no Capítulo 4. Nas seções seguintes, descreveremos os passos para alcançarmos os outros objetivos listados acima.

3.1.1 Genes Específicos e Ortólogos

Para apresentarmos os passos para encontrar os genes específicos e os genes ortólogos, necessitamos das seguintes definições:

- Sejam g e h genes dos proteomas G e H respectivamente;
- Seja $s(g, h)$ uma medida de significância estatística de similaridade de g e h, de tal modo que, quanto menor $s(g, h)$, mais similares g e h são. Sua implementação será descrita na Seção 3.2; e
- Seja A um alinhamento entre as seqüências representantes dos genes g e h. Sejam I_g, J_g, I_h, J_h posições de g e h como definidas abaixo:
  - I_g e J_g são o primeiro e o último símbolos de g que aparecem em A, respectivamente; e
  - I_h e J_h são o primeiro e o último símbolos de h que aparecem em A, respectivamente.

  A **cobertura** do alinhamento A em g , denotada por $c(A, g)$, é dada pelo percentual de $|g|$ que aparece em A. Assim,

$$c(A, g) = \frac{J_g - I_g + 1}{|g|} \times 100$$

  A mesma definição vale para $c(A, h)$, ou seja,

$$c(A, h) = \frac{J_h - I_h + 1}{|h|} \times 100$$

Segundo as definições acima, utilizamos o seguinte critério para a determinação dos genes ortólogos:

- Um gene h é **ortólogo** a um gene g e vice-versa se, e somente se :
  - $s(g, h) \leq S$, onde S é um limite fixo; e
  - o alinhamento A entre g e h é tal que $c(A, g) \geq P$ e $c(A, h) \geq P$, onde P é um limite fixo.

Quando os genes g e h são ortólogos e h é o gene de H que possui menor medida de significância estatística de similaridade com g, para qualquer gene $h^{'}$ de H ortólogo a g e vice-versa, definimos g e h como genes **fortemente ortólogos**. Utilizaremos essa definição de genes fortemente ortólogos objetivando o alinhamento entre os proteomas.

- Um gene g de G é **específico** em relação ao proteoma H se, e somente se, a medida de significância $s(g,h)$ é tal que $s(g,h) > S^{'}$ para qualquer gene h de H e $S^{'} \geq S$, onde $S^{'}$ é um limite fixo.

Conforme os critérios citados acima, necessitamos de um algoritmo que compare dois genes, fornecendo a similaridade e a significância estatística entre eles. Na Seção 3.2, descreveremos como essa comparação e como esses valores serão obtidos.

3.1.2 Regiões específicas (REs)

O problema de determinar REs pode ser modelado para o problema computacional conhecido como *subcadeia de máxima soma*, definido na Seção 2.2.

Na literatura, especificamente em análise de proteínas, encontramos algumas aplicações para esse problema, como a identificação de regiões de transmembranas e domínios de ligação, ambos citados em Ruzzo e Tompa [30].

Conforme definição do problema no Capítulo 2, utilizamos a mesma estratégia de Almeida, que consiste em atribuir valores aos genes do proteoma, onde um valor δ é atribuído para os genes não específicos e um valor Δ para os genes específicos, tal que $\Delta > \delta$. Assim, a seqüência de entrada para o problema é constituída pelos valores atribuídos aos genes.

Dessa forma, um algoritmo para encontrar todas as subcadeias maximais é suficiente para a implementação dessa estratégia. Na Seção 3.2 utilizaremos uma versão modificada do algoritmos de Ruzzo e Tompa [30], que foi descrita por Cáceres e colegas [8], e que resolve eficientemente esse problema.

3.1.3 Regiões Ortólogas (ROs)

Para descrevermos os métodos necessários para determinação das regiões ortólogas, as seguintes definições são necessárias:

Definição 3.1 (Run) *Sejam dois genomas G e H. Seja α uma RGC de G formada pelos genes $g_i, \ldots, g_k$ e β uma RGC de H formada pelos genes $h_j, \ldots, h_l$, tais que $k - i + 1 = l - j + 1$, $k > i$ e $l > j$. Dizemos que α e β formam um **run** se quaisquer das seguintes seqüências de pares de genes ortólogos ocorrerem:*

1. $(g_i, h_j), (g_{i+1}, h_{j+1}), \ldots, (g_k, h_l)$; *ou*
2. $(g_i, h_l), (g_{i+1}, h_{l-1}), \ldots, (g_k, h_j)$.

Um *run* é classificado como paralelo ou anti-paralelo. Classificamos um *run* como **paralelo** quando a seqüência de pares ortólogos corresponder a opção número 1 acima. Quando a seqüência de pares de ortólogos corresponder a opção número 2 acima, classificamos o *run* como **anti-paralelo**.

Os *runs* são classificados também como consistentes ou inconsistentes. Um *run* é classificado como **consistente** se, quando for paralelo, todos os pares de genes são tais que os genes participantes de cada par pertencem à mesma fita; e no caso de ser anti-paralelo, os genes de cada par pertencem a fitas opostas. Caso contrário, o *run* é classificado como **inconsistente**.

Assim, podemos redefinir o conceito de RO, descrito na Seção 3.1, detalhando as suas características.

Definição 3.2 (Região Ortóloga) *Definimos uma região ortóloga R como:*

1. *um run isolado com pelo menos M pares de ortólogos, onde M é um valor fixo; ou*
2. *a união de runs, cada um com um total de pelo menos M pares de ortólogos, e cuja distância entre os* ***genes extremos***[1] *de runs consecutivos não seja maior que um determinado valor fixo k, em número de genes; ou*
3. *um BBH. Definiremos o conceito de BBH na Seção 3.2.1.*

A estratégia para determinar ROs consiste em percorrer todos os runs, da esquerda para a direita (conforme a ordem dos genes de um dos proteomas), e juntar aqueles *runs* que são próximos, segundo o critério 2 acima definido.

Desta forma, a estratégia utilizada para a determinação das ROs está fundamentada na junção de *runs* próximos e na determinação de valores adequados para M e k.

Os testes realizados por Almeida em [1] sugerem que o valor $M = 3$, para genomas de procariotos, é suficiente para garantir que um *run* não seja encontrado ao acaso.

A implementação da estratégia para determinar as ROs será descrita detalhadamente na Seção 3.2.

3.1.4 Espinha dorsal dos proteomas

A estratégia aplicada para a determinação da espinha dorsal entre dois proteomas consiste no alinhamento global entre eles. O alinhamento entre os proteomas é baseado no problema computacional denominado *subseqüência comum mais longa* [14].

Particularmente, cada símbolo do alinhamento corresponde ao número seqüencial do gene no proteoma, e dois símbolos das cadeias são iguais se, e somente se, os respectivos genes compartilham uma determinada relação.

Para minimizar a interferência de genes parálogos, a relação exigida para que dois genes (g, h) sejam candidatos a se alinharem, é que g e h devem ser genes fortemente ortólogos.

[1] Os genes extremos dos *runs* são aqueles genes que encontram-se mais próximos de um outro *run*.

Dessa forma, objetivamos obter o alinhamento com maior número de pares de genes ortólogos, sem que existam cruzamentos, o que caracteriza um alinhamento.

Podemos observar, nesse caso, a diferença entre a estratégia acima definida, e a estratégia utilizada para a determinação de *runs*. No caso da espinha dorsal, aplicamos o conceito de ortologia forte para minimizarmos a ação dos genes parálogos. Para a determinação dos *runs*, não podemos aplicar a mesma estratégia, pois um determinado gene pode participar de mais de um par de ortólogos, podendo participar de mais de uma região ortóloga, devido as duplicações internas que podem ocorrer no genoma [34].

A seção seguinte contém as implementações e descrições da mais detalhadas da utilização dessa metodologia para determinação da espinha dorsal, que procura evidenciar a proximidade dos proteomas.

3.2 Nova implementação

Nesta seção abordaremos uma nova proposta de implementação para a metodologia de Almeida, descrita anteriormente. Inicialmente, a primeira implementação da metodologia resultou no programa de computador denominado de ***E**xtended **G**enome-**G**enome comparsion* (EGG). EGG foi inicialmente formulado por Almeida e Setubal em [3], posteriormente reformulado em [4] e em [1].

Posteriormente à última reformulação, existiu a necessidade de uma reestruturação do programa, que atendesse às correções de pequenas falhas, a melhoria da portabilidade e desempenho, além da inclusão de novas funcionalidades e disponibilização para os usuários. Conforme essas necessidades, descreveremos nesse capítulo a terceira reformulação de EGG.

Segundo Almeida [1], o objetivo principal de EGG é a comparação de dois proteomas. Primeiramente, é realizada uma comparação das proteínas preditas, na forma *todas-contra-todas*, utilizando o BLAST. Posterior a essa comparação, um grafo bipartido é construído, onde o conjunto de vértices é constituído pelos genes particionados pelos genomas e o conjunto de arestas é constituído pelas ortologias entre os genes. Por fim, algumas estruturas organizacionais são construídas para alcançar os objetivos listados na Seção 3.1.

Assim como a metodologia, esta nova reimplementação também é baseada na reformulação apresentada por Almeida em [1].

3.2.1 Descrição das fases de EGG

Podemos distinguir em EGG três fases importantes:

- Comparação dos genes *todos-contra-todos*;
- Construção do grafo bipartido; e
- Determinação das estruturas organizacionais.

Nas seções subseqüentes, descreveremos cada uma das 3 fases de EGG e detalhes de suas implementações.

Comparação dos genes *todos-contra-todos*

Nessa fase, temos a comparação de cada gene g_i de um genoma G contra todos os genes do genoma H e, em seguida, cada gene h_j de H é comparado contra todos os genes de G. O objetivo consiste em relacionar os genes com a finalidade de determinar os genes ortólogos e genes específicos.

Para realizar esse procedimento, EGG utiliza como ferramenta comparativa o programa BLAST [6, 7]. Esse programa fornece para cada gene g_i, uma lista de genes h_j similares a g_i, dentre todos os genes de H e algumas informações sobre os alinhamentos entre esses genes. Cada gene similar h_j retornado por BLASTé denominado de **hit**.

Para implementar o valor de similaridade descrito na Seção 3.1.1, usamos a medida de significância estatística do BLAST, denominada *e-value*. Conforme descrito no Capítulo 2, essa medida é proporcional à probabilidade de um determinado *hit* ter sido encontrado ao acaso. Assim, quanto menor é o valor de *e-value*, menor é a probabilidade de um *hit* ter sido encontrado ao acaso, ou seja, mais significante é o *hit*.

Para armazenarmos os *hits* obtidos pelo BLAST, utilizamos a estrutura de dados lista ligada, que armazenará para cada gene g_i de G uma lista de todos os seus *hits* h_j de H, ordenados de forma não-decrescente pelo valor de *e-value*. Esta forma de armazenamento permite obter o melhor *hit* de um gene g_i em tempo constante.

Ao final dessa fase, EGG consegue determinar os genes específicos entre os proteomas, objetivo 1 listado na Seção 3.1. Especificamente, EGG considera que um gene g_i é específico em relação a H, se g_i não obteve *hits* com *e-value* menor ou igual a $S' = 10^{-3}$. Na seção seguinte descreveremos o critério para encontrarmos os genes ortólogos.

Determinação das arestas do grafo

Na última reformulação, EGG criou um grafo bipartido determinando as ortologias entre os genes dos genomas. Essas ortologias são estabelecidas através da especificação do relacionamento, denominado *match*, entre os genes. EGG utiliza os *matches* como arestas do grafo bipartido. Utilizaremos o termo *match* no lugar de "par de genes ortólogos" no decorrer do texto.

Um ***match*** entre os genes g_i de G e h_j de H ocorre quando g_i encontrar h_j como *hit*, com os seguintes valores limites: $S = 10^{-5}$, $P = 60$ e vice-versa. Esses valores foram utilizados por Almeida em [1]. Tamames [39] utilizou os valores 10^{-5} e 75 para S e P respectivamente. Assim como em Almeida, os valores de S e P podem ser alterados pelo usuário.

Pela determinação dos *matches*, EGG atinge o objetivo 3, que é determinar pares de genes ortólogos. Assim, podemos reescrever os seguintes critérios para obtermos os genes ortólogos e os genes específicos:

- Dados dois genes g e h pertencentes, respectivamente, aos genomas G e H, e os alinhamentos A e $A^{'}$ retornado pelo BLAST, tais que A é o alinhamento de g para h e $A^{'}$ é o alinhamento de h para g. Dizemos que g e h são ortólogos se, e somente se $(s(g,h)+s(h,g))/2 \leq 10^{-5}$ e $(c(A,g)+c(A^{'},g))/2 \geq 60$ e $(c(A,h)+c(A^{'},h))/2 \geq 60$.
- Um gene g é específico em relação a um genoma H se, e somente se, $s(g,h_j) > 10^{-3}$, para $1 \leq j \leq |H|$.

Conforme os critérios acima definidos, consideramos que os *e-values* menores ou iguais a 10^{-5} indicam homologia com alta probabilidade. Dessa forma, dois genes são considerados ortólogos se eles forem homólogos com alta probabilidade e se a cobertura do alinhamento entre eles for maior ou igual a $P = 60$. Por outro lado, os genes que não possuírem *hits* com *e-values* menores ou iguais a 10^{-3} são denominados como genes específicos. Por esses critérios, consideramos a região compreendida entre 10^{-3} e 10^{-5} como uma região de dúvida, da terminologia em inglês *"twilight zone"*.

A implementação do conceito de "pares de genes fortemente ortólogos" é realizada através da utilização do melhor *hit* bidirecional, da tradução do termo em inglês, ***B**idirectional **B**est **H**it* (**BBH**). O termo BBH foi empregado em [1, 4, 35, 39] também para determinar os pares de genes ortólogos. Um par de genes ortólogos (g_i, h_j) formam um BBH, se h_j é o melhor *hit* encontrado por g_i, ou seja, com menor *e-value*, e vice-versa [1].

Armazenamos os *matches* entre os genes g_i de G e h_j de H na mesma estrutura que implementa um *hit* h_j de um gene g_i, sinalizando quando (g_i, h_j) formar um *match*. Como os *hits* de um gene g_i estão armazenadas na lista ligada de *hits*, com o melhor *hit* de g_i na primeira posição da lista, em tempo constante, obtemos o melhor *hit* de um determinado gene g_i. Dessa forma, obtemos todos os BBHs de dois proteomas em tempo linear no número de genes de G e H.

Determinação das estruturas organizacionais.

Nessa fase descreveremos as regiões específicas, os runs, as regiões ortólogas e a espinha dorsal entre dois proteomas.

Regiões específicas

A implementação das regiões específicas é realizada conforme a estratégia descrita na Seção 3.1.2. Definimos que o valor $\delta = -1$ é atribuído para os genes não específicos e $\Delta = 1$ para os genes específicos, atendendo a restrição $\Delta > \delta$. Dessa forma, faz-se necessária a implementação de um algoritmo que encontre todas as subseqüências contíguas de soma máxima, de uma seqüência de entrada A composta pelos valores 1 e -1.

Segundo Ruzzo e Tompa [30], as características do problema de determinar todas as subseqüências de soma máxima de uma seqüência X sugerem um algoritmo simples de divisão e conquista com os seguintes passos:

1. Encontre a subseqüência máxima de maior soma e remova-a da seqüência X;

2. Aplique o algoritmo recursivamente, para as partes restantes à esquerda e à direita da porção removida.

No entanto, a análise desse algoritmo é similar a análise do algoritmo de ordenação *Quick-Sort*, que no pior caso, necessitará de tempo quadrático para encontrar a solução do problema [14].

No mesmo trabalho [30], Ruzzo e Tompa descrevem um algoritmo para encontrar todas as subseqüências de máxima soma em tempo linear. Segundo os próprios autores, o algoritmo, conforme descrito, não executa em tempo linear, sendo necessária uma alteração em uma possível fase de implementação, para que o tempo total de execução do algoritmo, utilizando a análise amortizada, torne-se linear.

Inicialmente, implementamos a versão de divisão e conquista conforme descrita em [30]. Devido aos testes de comparação de tempo, entre os algoritmos da versão de divisão e conquista e de tempo linear, realizados por Ruzzo e Tompa em [30]; e pela diferença entre suas complexidades, decidimos implementar o algoritmo descrito por Alves e colegas [8], que é uma versão modificada do algoritmo de Ruzzo e Tompa, que também possui complexidade $O(n)$ amortizada.

Embora os algoritmos de [30] e [8] possuam complexidades similares, o algoritmo apresentado por Alves [8] está descrito de forma mais explícita, fazendo utilização de vetores com a finalidade de facilitar a análise e compreensão; e também mantendo a mesma complexidade da versão do algoritmo de Ruzzo e Tompa. Neste algoritmo, a entrada é uma seqüência A e a saída são dois vetores denominados $Mlista(A)$ ($Ml(A)$) e $Plista(A)$ ($Pl(A)$), que armazenam, respectivamente, as informações sobre as subseqüências maximais e o seu índice de ocorrência na seqüência A. Intuitivamente, o algoritmo mantém o vetor $Ml(A)$ com as informações de cada subseqüência de máxima pontuação e um vetor de índices de subseqüências candidatas de máxima soma prefixa, onde pretende-se estender alguma subseqüência candidata para que transforme-se em uma subseqüência maior e de máxima soma. O algoritmo descrito por Alves e colegas utiliza os vetores $Pl(A)$ e $Ml(A)$ como a estrutura de dados de pilhas. Descrevemos em seguida, de forma sucinta, o algoritmo Subseqüências-Maximais de Alves e colegas.

Algoritmo 1 SUBSEQÜÊNCIAS-MAXIMAIS

Entrada: Seqüência $A = (a_1, a_2, \ldots, a_{|A|})$

Saída: $Ml(A)$ e $Pl(A)$ com n_m e n_p elementos. s mantém a soma de cada subseqüência.

1: $n_m \leftarrow 0$, $n_p \leftarrow 0$, $s \leftarrow 0$
2: **para** $i \leftarrow 1$ **até** $|A|$ **faça**
3: $s \leftarrow s + a_i$
4: **se** a_i negativo **então**
5: **enquanto** tiver subseqüência candidata **e** ela não contribuir para a soma da subseqüência atual **faça**
6: $n_p \leftarrow n_p - 1$ {Desempilha subseqüência candidata}
7: **fim enquanto**
8: **fim se**
9: **se** a_i positivo **então**
10: {Empilha a nova seqüência formada por a_i}
11: $n_m \leftarrow n_m + 1$
12: {Obtém informações da nova seqüência}
13: {Armazena índice n_m no vetor $Pl(A)$}
14: **enquanto** tiver subseqüência candidata **e** ela não contribuir para a extensão até a_i **faça**
15: $n_p \leftarrow n_p - 1${Desempilha subseqüência candidata}
16: **fim enquanto**
17: $n_p \leftarrow n_p + 1$
18: {$Pl[n_p]$ aponta para a melhor subseqüência candidata}
19: $n_m \leftarrow Pl[n_p]$
20: {Completa as informações da melhor subseqüência}
21: **fim se**
22: **fim para**

Todo comando presente no laço na linha 2 executa em tempo constante, exceto os laços das linhas 5 e 14. Nesse caso, com a análise amortizada no número de elementos da pilha $Pl(A)$, observamos que os laços não irão procurar por todas as subseqüências de A, mas apenas pelas candidatas de máxima soma. Por meio da análise amortizada temos que o custo do algoritmo é $O(n)$ amortizado. As provas da complexidade amortizada do algoritmo, assim como maiores detalhes da implementação da versão de Alves e colegas estão descritos em [8].

Em nosso trabalho particularmente, a entrada do algoritmo é o vetor A constituído pelos valores conforme definidos anteriormente. Porém, a saída do algoritmo são as subcadeias A_i de máxima soma, do vetor de entrada A, tal que $|A_i| \geq w$. Nesse trabalho utilizamos o valor $w = 10$, que pode ser alterado pelo usuário.

Como saída de EGG, temos um arquivo texto onde as REs encontradas são apresentadas. Na Figura 3.4 temos uma RE de *Xanthomonas axonopodis pv. citri str. 306* (`Xac`) em relação ao proteoma de *Xanthomonas campestris pv. campestris str. ATCC 33913* (`Xcc`). Maiores detalhes sobre o arquivo podem ser vistos em no Apêndice A.

```
>Region from 4061 to 4091
31 orfs
=====================================================================
  Gene   Synonym        start..end        product
=====================================================================
# -_      XAC4118     4833465..4833995    hypothetical protein
# -_      XAC4119     4833998..4837528    hypothetical protein
# -_      XAC4120     4837525..4838880    hypothetical protein
# -_      XAC4121     4838787..4840121    hypothetical protein
# -_      XAC4122     4840701..4842017    hypothetical protein
# -_      XAC4123     4842304..4842852    hypothetical protein
# -_      XAC4124     4842849..4844885    hypothetical protein
  -_      XAC4125     4844967..4846298    hypothetical protein
# -_      XAC4126     4846534..4846929    hypothetical protein
# -pknB   XAC4127     4846892..4849189    serine threonine kinase
# -ecfR   XAC4128     4849390..4849929    extracytoplasmic sigma factor
  +rpoE   XAC4129     4850278..4850805    ECF sigma factor
  +_      XAC4130     4850802..4851872    transmembrane_sensor
  +_      XAC4131     4852271..4855222    hypothetical protein
  +appA   XAC4132     4855361..4856611    6 phytase
# +_      XAC4133     4856619..4857986    hypothetical protein
# +_      XAC4134     4858039..4858398    hypothetical protein
# -_      XAC4135     4858635..4859336    hypothetical protein
# -_      XAC4136     4859777..4861771    hypothetical protein
  +_      XAC4137     4862893..4864107    ISxac1 transposase
  +_      XAC4138     4864112..4864522    transposase
# -_      XAC4139     4864528..4865544    hypothetical protein
  -clpB   XAC4140     4865531..4868311    ClpB
# -_      XAC4141     4868367..4869407    hypothetical protein
# -_      XAC4142     4869371..4871254    hypothetical protein
# -_      XAC4143     4871259..4871762    hypothetical protein
# -_      XAC4144     4871768..4872607    hypothetical protein
# -_      XAC4145     4872761..4873264    hypothetical protein
# -_      XAC4146     4873345..4874838    hypothetical protein
# -_      XAC4147     4874842..4875351    hypothetical protein
# +feaR   XAC4148     4875665..4876336    transcriptional regulator
```

Figura 3.4: Exemplo de região específica de *Xanthomonas axonopodis pv. citri str. 306* em relação a *Xanthomonas campestris pv. campestris str. ATCC 33913*. O símbolo # indica os genes que pertencem a região específica.

Runs

Conforme descrito na metodologia, para obter as regiões ortólogas, devemos determinar previamente os *runs*. A implementação dos *runs* é realizada utilizando os *matches*, conforme descrito na Seção 3.2.1. Segundo a Definição 3.1, temos que um *run* é uma seqüência de pelo menos dois *matches*. Dessa forma, determinamos os *runs* primeiramente armazenando os *matches* em uma matriz binária $\mathcal{A}_{mn}$, onde m é o número de genes do genoma G e n o número de genes do genoma H, tal que $\mathcal{A}_{i,j} = 1$ se, e somente se, os genes g_i de G e h_j de H formam um *match*. Em seguida, percorremos a matriz $\mathcal{A}$ procurando por pelo menos duas posições consecutivas em qualquer diagonal, onde as posições estão preenchidas com 1. Em seguida descrevemos, em pseudo-código, o algoritmo Constrói-Run que determina os *runs*.

Algoritmo 2 CONSTRÓI-RUN

Entrada: Uma matriz binária A_{mn}.
Saída: Uma lista de runs

```
1: para i ← m até 1 faça
2:   para j ← n até 1 faça
3:     {Obtém as coordenadas finais do run.}
4:     k ← i
5:     l ← j
6:     enquanto A[k][l] ≠ 0 faça
7:       {Obtém informações sobre consistência.}
8:       k ← k − 1
9:       l ← l − 1
10:    fim enquanto
11:    {Obtém as coordenadas iniciais e o código do run.}
12:  fim para
13: fim para
```

Conforme a definição de *run* na Seção 3.1, temos que os *runs* podem ser anti-paralelos. Nesse caso, o algoritmo é análogo ao acima apresentado, com alterações no índice da linha 2 e nas linhas 8 e 9. O custo do algoritmo descrito acima, no melhor caso, é $O(mn)$, que ocorre quando todos os elementos da matriz forem $0s$. Por outro lado, o custo de pior caso é $O(mn)^2$ e ocorre quando todos os elementos da matriz forem $1s$.

O programa EGG apresenta em um arquivo texto os *runs* encontrados, onde cada *run* é identificado por um código. O código de um *run* é composto pelas seguintes informações: os seis primeiros símbolos identificam os pares de genomas comparados; os 8 dígitos seguintes identificam o ano, o mês e o dia; o número seguinte é um número seqüencial do *run* na comparação proteômica; e os caracteres finais indicam se o *run* é paralelo ou anti-paralelo e se é consistente ou anti-consistente.

Nas Figuras 3.5 e 3.6 temos, respectivamente, um trecho do arquivo resultante da comparação entre *Xanthomonas axonopodis pv. citri str. 306* e *Xanthomonas campestris pv. campestris str. ATCC 33913*; e uma representação gráfica desse *run*. Maiores detalhes sobre o arquivo texto estão apresentados no Apêndice A.

```
>XACXCC20060329-245-Pc
# of matches: 5
6kb in XAC - 6kb in XCC
=============================================================================================
 Gene  |Synonym     start size e-value [       best hit        ] product
=============================================================================================
-_     |XAC0925   1087413 139    3e-43 [best                   ] hypothetical protein
-_     |XCC0848   1007675 139    4e-63 [best                   ] hypothetical protein
---------------------------------------------
+_     |XAC0926   1088018 184   1e-103 [best                   ] hypothetical protein
+_     |XCC0849   1008268 184   1e-103 [best                   ] hypothetical protein
---------------------------------------------
+ilvE  |XAC0927   1088637 361         0 [best                   ] branched chain amino acid aminotransferase
+ilvE  |XCC0850   1008887 361         0 [best                   ] branched chain amino acid aminotransferase
---------------------------------------------
+_     |XAC0928   1090032 575         0 [best                   ] extracellular protease
+_     |XCC0851   1010286 580         0 [best                   ] extracellular protease
---------------------------------------------
+_     |XAC0929   1091789 546   1e-162 [+_     |XCC0851 /1e-174] extracellular protease
+_     |XCC0852   1012120 518   1e-169 [best                   ] extracellular protease
---------------------------------------------
```

Figura 3.5: Exemplo de um *run* paralelo consistente entre *Xanthomonas axonopodis pv. citri str. 306* e *Xanthomonas campestris pv. campestris str. ATCC 33913*.

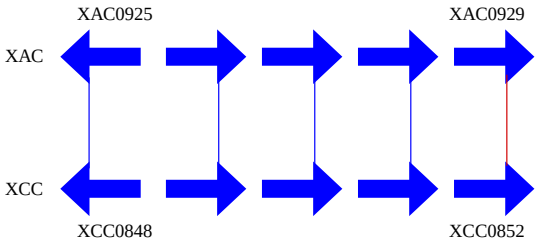


Figura 3.6: Representação gráfica do *run* do trecho de arquivo da Figura 3.5

Na figura acima, as setas para a esquerda representam os genes pertencentes a fita `'-'`, enquanto que as setas para a direita representam os genes pertencentes a fita `'+'`. As linhas de cor azul conectam os genes que fizeram BBH, enquanto que a linha de cor vermelha conecta os genes que fizeram *match*.

Regiões ortólogas

Segundo a Definição 3.1.3, uma região ortóloga é composta por M pares de ortólogos ou pela união de *runs* com pelo menos M pares de ortólogos que distam, a partir dos seus genes extremos, no máximo um determinado valor fixo k.

Como a metodologia está baseada na união de *runs* próximos, devemos implementar a noção de distância entre os genes extremos dos *runs*. Descreveremos abaixo uma implementação, conforme Almeida [1], para determinarmos a proximidade adequada para juntar os *runs*.

Sejam R_1 e R_2 dois *runs* entre os genomas G e H. Sem perda de generalidade, segundo a definição de *run* da Seção 3.1, representamos R_1 e R_2 como:

$$R_1 = (g_i, h_j), (g_{i+1}, h_{j+1}), \ldots, (g_k, h_l) \text{ e } R_2 = (g_p, h_q), (g_{p+1}, h_{q+1}), \ldots (g_r, h_s)$$

Sejam também I_G e I_H os números de genes entre os *runs* nos proteomas G e H, respectivamente, tal que $I_G = p - k - 1$ e $I_H = q - l - 1$; $I_{\min}$ e $I_{\max}$ os intervalos mínimos e máximos entre os *runs*; **max_small_gaps** e **max_large_gaps** os valores dos tamanhos máximos, do menor e do maior intervalo entre os *runs*, fornecidos pelos usuário.

Como queremos juntar os *runs* próximos formando uma só região que evidencie um bloco de genes com certo grau de ortologia [1], juntaremos os *runs* conforme a seguinte regra de distância:

$$I_{\min} \leq \textbf{max\_small\_gap} \text{ e } I_{\max} \leq \textbf{max\_large\_gap}$$

Na figura 3.7, temos uma representação gráfica de uma junção entre dois *runs* R_1 e R_2, seguindo as restrições da regra de distância descrita acima. Nesse caso, definimos **max_small_gap** $= 5$ e **max_large_gap** $= 2$.

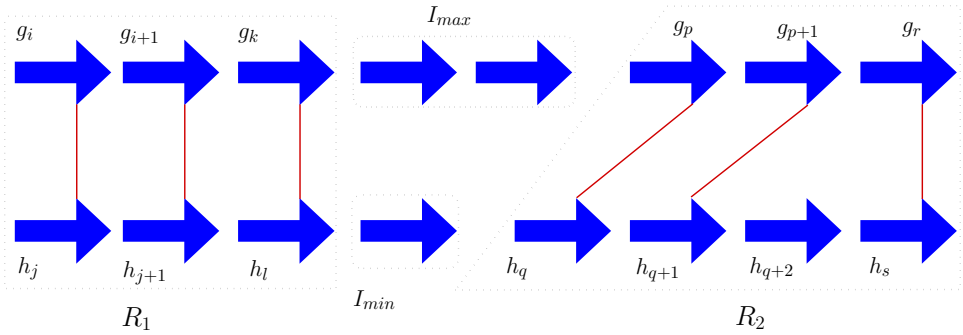


Figura 3.7: Exemplo de uma junção entre dois *runs* R_1 e R_2.

Denominamos os *runs* que obedecem a relação de distância, de *runs* **próximos**, e o procedimento de união dos *runs* de **junção**.

Utilizamos o algoritmo incremental de Almeida [1] para determinar as ROs entre dois proteomas. Abaixo temos uma descrição de um pseudo-código para esse algoritmo.

Algoritmo 3 JUNTA-RUNS

Entrada: LR : uma lista de runs
Saída: LRO : uma lista de ROs

```
1: LRO ← ∅
2: para i ← 1 até |LR| faça
3:    para j ← i + 1 até |LR| faça
4:       {Obtém I_min e I_max}
5:       se I_min ≤ max_small_gaps e I_max ≤ max_large_gaps então
6:          {junta os runs i e j}
7:       fim se
8:    fim para
9: fim para
```

O algoritmo, de forma incremental, realiza a junção dos *runs* do início para o final dos proteomas, ou seja, uma região ortóloga resultante da união de *runs* próximos poderá ser unida com o próximo *run* à direita. A complexidade de tempo do algoritmo, no pior caso é $O(|LR|^2)$, onde $|LR|$ é o número de *runs* na lista LR. O pior caso do pesudo-algoritmo ocorre quando todos os *runs* não passarem na regra de distância, definida anteriormente.

Segundo Almeida, poderemos ter uma situação onde um *run* está próximo a um *match* isolado, e ambos não podem ser juntados se estiverem isolados no decorrer do proteoma. Essa junção pode ser importante, pois poderá gerar uma região com 3 ou mais *matches*. Para esses casos, permitiremos que o *run* possa ser juntado com o *match* se este *match* contribuir significativamente para a região, ou seja, se o *match* for BBH e obedecer a regra de distância em relação ao *run*. Logo, consideramos um BBH isolado como um *run*.

EGG mostra as regiões ortólogas encontradas em um arquivo texto. Na Figura 3.8 temos um trecho do arquivo texto da Região Ortõloga resultante da comparação entre *Xylella fastidiosa 9a5c* e *Neisseria meningitidis MC58*. Maiores detalhes sobre o arquivo de texto podem ser vistos no Apêndice A.

```
>XFNMB20060710-26-Rc
7 matches
7kb in XF - 10kb in NMB
=====================================================================
Gene   Synonym (XF)       gi        size    product
=====================================================================
+_     XF0736          15837338    635aa    threonyl-tRNA synthetase
+infC  XF0737          15837339    159aa    translation initiation factor IF-3
+_     XF0738          15837340     31aa    hypothetical protein
+rpmI  XF0739          15837341     65aa    50S ribosomal protein L35
+_     XF0740          15837342    119aa    50S ribosomal protein L20
+pheS  XF0741          15837343    333aa    phenylalanyl-tRNA synthetase alpha subunit
+pheT  XF0742          15837344    792aa    phenylalanyl-tRNA synthetase beta subunit
+_     XF0743          15837345     99aa    integration host factor alpha subunit
=====================================================================
Gene   Synonym (NMB)      gi        size    product
=====================================================================
+thrS  NMB0720         15676618    637aa    threonyl-tRNA synthetase
+infC  NMB0721         15676619    155aa    translation initiation factor 3
+rpmI  NMB0722         15676620     65aa    50S ribosomal protein L35
+rplT  NMB0723         15676621    119aa    50S ribosomal protein L20
+pheS  NMB0724         15676622    330aa    phenylalanyl-tRNA synthetase alpha subunit
+_     NMB0725         15676623    352aa    modification methylase HgaI-1
+_     NMB0726         15676624    489aa    type II restriction enzyme HgaI
+_     NMB0727         15676625    216aa    N-6 adenine-specific DNA methylase
+phe   NMB0728         15676626    787aa    phenylalanyl-tRNA synthetase beta subunit
+himA  NMB0729         15676627    100aa    integration host factor, alpha subunit
=======
matches
=======
=============================================================================================
 Gene   Synonym    start size e-value [       best hit       ] product
=============================================================================================
+_      XF0743    698556 99     1e-27 [best                  ] integration host factor alpha subunit
+himA   NMB0729   761371 100    2e-27 [best                  ] integration host factor, alpha subunit
---------------------------------------------
+pheT   XF0742    696154 792   1e-151 [best                  ] phenylalanyl tRNA synthetase beta subunit
+pheT   NMB0728   758934 787   1e-150 [best                  ] phenylalanyl tRNA synthetase beta subunit
---------------------------------------------
+pheS   XF0741    695069 333   1e-104 [best                  ] phenylalanyl tRNA synthetase alpha subunit
+pheS   NMB0724   754557 330    2e-88 [best                  ] phenylalanyl tRNA synthetase alpha subunit
---------------------------------------------
+_      XF0740    694438 119    2e-40 [best                  ] 50S ribosomal protein L20
+rplT   NMB0723   753852 119    1e-35 [best                  ] 50S ribosomal protein L20
---------------------------------------------
+rpmI   XF0739    694230 65     2e-12 [best                  ] 50S ribosomal protein L35
+rpmI   NMB0722   753642 65     3e-12 [best                  ] 50S ribosomal protein L35
---------------------------------------------
+infC   XF0737    693490 159    2e-47 [best                  ] translation initiation factor IF 3
+infC   NMB0721   753028 155    3e-55 [best                  ] translation initiation factor 3
---------------------------------------------
+_      XF0736    691467 635       0 [best                  ] threonyl tRNA synthetase
+thrS   NMB0720   751043 637       0 [best                  ] threonyl tRNA synthetase
---------------------------------------------
```

Figura 3.8: Exemplo de uma RO entre *Xylella fastidiosa 9a5c* e *Neisseria meningitidis MC58*.

Na Figura 3.9, temos uma representação gráfica da região ortóloga da Figura 3.8 resultante da junção de 3 runs com 2, 3 e 2 matches respectivamente. O gene de cor preta ilustra um gene anotado como hipotético. O sentido das setas representam as orientações dos genes, como na Figura 3.6.

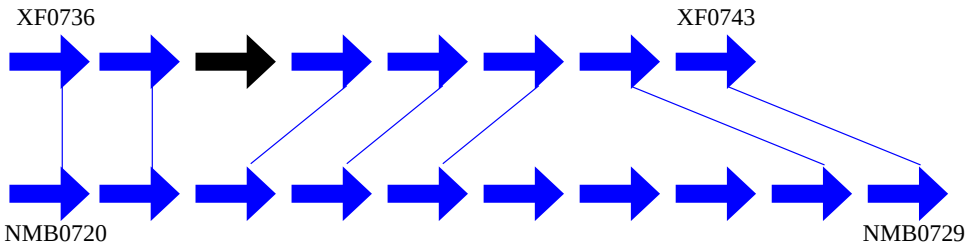


Figura 3.9: Representação gráfica da RO da Figura 3.8.

Espinha Dorsal

Segundo a metodologia descrita na Seção 3.1.4, EGG implementa a espinha dorsal entre os proteomas utilizando o algoritmo de programação dinâmica descrito por Cormen e outros [14], para o problema de LCS, definido na Seção 2.2. Particularmente, as seqüências s e t de entrada são tais que, $s_i = i$, para $1 \leq i \leq m$, representando os genes de G e $t_j = p(j)$, para $1 \leq j \leq n$, representando os genes de H, onde $p(j) = i$ se, e somente se, (g_i, h_j) forem BBHs, ou $p(j) = 0$ caso contrário.

O programa EGG encontra a espinha dorsal entre os proteomas de forma direta e reversa, com a finalidade de encontrar a espinha dorsal que mais evidencie o quanto os genomas são parecidos. Na forma direta, as seqüências s e t são conforme definimos acima. Porém na forma reversa, trocamos a seqüência t pela sua seqüência reversa e comparamos com a seqüência s da mesma forma como descrito no parágrafo acima.

Por fim, EGG mostra as espinhas dorsais em arquivos textos. Na figura 3.10 temos um trecho do arquivo texto que mostra a espinha dorsal direta entre os proteomas de *Xanthomonas axonopodis pv. citri str. 306* e *Xanthomonas campestris pv. campestris str. ATCC 33913*. Maiores detalhes sobre o arquivo texto podem ser vistos no Apêndice A.

```
=============================================================================================================================================
                          PRODUCT  START..END    GENE(STRAND)        (STRAND)GENE   START..END    PRODUCT
=============================================================================================================================================
  chromosomal replication initiator     42..1370   XAC0001 (+)  <<<>>>  (+) XCC0001     42..1370   chromosomal replication initiator
      DNA polymerase III beta chai   1647..2747   XAC0002 (+)  <<<>>>  (+) XCC0002    1646..2746   DNA polymerase III beta chain
    DNA replication and repair RecF   3799..4905   XAC0003 (+)  <<<>>>  (+) XCC0003    3633..4739   DNA replication and repair RecF protein
                DNA gyrase subunit   5020..7464   XAC0004 (+)  <<<>>>  (+) XCC0004    4853..7297   DNA gyrase subunit B
              hypothetical protein   7685..8368   XAC0005 (+)  <<<>>>  (+) XCC0005    7359..8201   hypothetical protein
              hypothetical protein   8552..9358   XAC0006 (+)  <<<>>>  (+) XCC0006    8264..9070   hypothetical protein
              hypothetical protein   9636..10829  XAC0007 (+)  <<<>>>  (+) XCC0007    9209..10405  hypothetical protein
                     TonB protein  10983..11654  XAC0008 (+)  <<<>>>  (+) XCC0008   10559..11230  TonB protein
  biopolymer transport ExbB protein  11740..12501  XAC0009 (+)  <<<>>>  (+) XCC0009   11315..12076  biopolymer transport ExbB protein
biopolymer transport ExbD1 protein  12548..12970  XAC0010 (+)  <<<>>>  (+) XCC0010   12123..12545  biopolymer transport ExbD1 protein
biopolymer transport ExbD2 protein  12974..13387  XAC0011 (+)  <<<>>>  (+) XCC0011   12549..12959  biopolymer transport ExbD2 protein
   pyridoxal phosphate biosynthetic  13649..14416  XAC0012 (-)  <<<>>>  (-) XCC0012   14113..14883  pyridoxal phosphate biosyn
              hypothetical protein  14424..14756  XAC0013 (-)  <<<>>>  (-) XCC0013   14891..15160  hypothetical protein
            cardiolipin synthetas  14768..16228  XAC0014 (-)  <<<>>>  (-) XCC0014   15235..16695  cardiolipin synthetase
              hypothetical protein  16671..17330  XAC0015 (+) #            -
              hypothetical protein  17330..17920  XAC0016 (+) #            -
              hypothetical protein  18131..19258  XAC0017 (-)  <<<>>>  (+) XCC0015   16981..18075  hypothetical protein
              hypothetical protein  19442..20359  XAC0018 (-)  <<<>>>  (-) XCC0016   18170..18940  hypothetical protein
            outer_membrane protein  20413..21753  XAC0019 (-)  <<<>>>  (-) XCC0017   19513..20853  outer membrane protein
              hypothetical protein  21972..22664  XAC0020 (+)  <<<>>>  (+) XCC0018   21074..21766  hypothetical protein
```

Figura 3.10: Trecho do arquivo texto da espinha dorsal direta entre os BBHs dos genomas dos organismos *Xanthomonas axonopodis pv. citri str. 306* (Xac) e *Xanthomonas campestris pv. campestris str. ATCC 33913* (Xcc).

3.3 Novas Funcionalidades

Com o advento do seqüenciamento de ESTs, muitas das ESTs tem sido seqüenciadas como uma alternativa ao seqüenciamento completo dos genomas. Ferramentas de Bioinformática baseadas em análise de seqüências têm sido estendidas ao escopo da análise de ESTs no campo da proteômica, desenvolvimento de marcadores e anotação genômica [29].

Embora existam metodologias baseadas em *arrays* (*Macroarrays* ou *Microarrays*) que permitem a investigação massiva e de forma paralela da expressão de genes, podemos utilizar também as seqüências de ESTs para inferir similaridades entre genes e ESTs, como realizado com ESTs de fungos patógenos e genes de fungos não patógenos [41].

Dessa forma, para utilizarmos o programa EGG para análise ou comparação entre ESTs e genes de genomas seqüenciados completamente, necessitaríamos manipular as informações de entrada (ESTs) para que EGG executasse adequadamente. Assim, surgiu a necessidade da implementação de uma ferramenta menos robusta, porém que pudesse inferir informações sobre os conjuntos de seqüências em comparação. Para atender a essas necessidades, implementamos a ferramenta denominada EGG-LITE, que possibilita a comparação de dois conjuntos de seqüências provenientes de genomas incompletos para inferir similaridade entre elas.

A estrutura do programa EGG-LITE é semelhante a EGG, porém sem a criação das estruturas organizacionais, terceira fase de EGG. EGG-LITE realiza a comparação das seqüências de forma todas-contra-todas, utilizando também a ferramenta BLAST. No final dessa fase, EGG-LITE, assim como EGG, determina os genes específicos, da mesma forma como descrito na Seção 3.2.1. Em seguida, os matches e os BBHs também são determinados,

conforme descrito na Seção 3.2.1. Como esperado, EGG-LITE não constrói as estruturas organizacionais, pois estas dependem de informações relativas ao posicionamento dos genes no genoma.

O programa EGG-LITE mostra a descrição das seqüências que fizeram BBH, *matches* e que são específicas, em arquivos texto, contendo as informações descritivas de cada seqüência a partir de seus arquivos multi-fasta.

As figuras a seguir mostram trechos dos arquivos de saída para as seqüências que fizeram BBH, *matches* e que são específicas.

```
##############################
CGPB bidirectional best hits
##############################

================================================================================================
 Identifier
================================================================================================
(CHG00002.1) hypothetical protein (translation)
Contig1420 nucleotide excision repair protein rad23 homolog
------------------------------------
(CHG00007.1) hypothetical protein (translation)
PBGAC-M1-015t_D06 Sulfur metabolite repression control protein
------------------------------------
(CHG00008.1) hypothetical protein (translation)
Contig1600
------------------------------------
(CHG00009.1) hypothetical protein (translation)
PBDEX-M1-035t_D07 zinc metallo-protease
------------------------------------
(CHG00013.1) hypothetical protein (translation)
Contig582  vacuolar aminopeptidase ysc1
------------------------------------
```

Figura 3.11: Trecho do arquivo de BBHs entre genes do organismo *Chaetomium globosum* (Cg) e ESTs do organismo *Paracoccidioides brasiliensis* (Pb), respectivamente.

```
##############################
 matches between PA and PB
##############################

================================================================================================
 Identifier
================================================================================================
pans_AUG:pans_0-g1.1 pans_0:join(684..752,818..1464,1517..2027) cdslen=1227
Contig25 Probable 26S protease subunit and member of CDC48/PAS1/SEC18 family of ATPases
------------------------------------
pans_AUG:pans_0-g1.1 pans_0:join(684..752,818..1464,1517..2027) cdslen=1227
Contig935 Microsomal protein of CDC48/PAS1/SEC18 family of ATPases
------------------------------------
pans_AUG:pans_0-g1.1 pans_0:join(684..752,818..1464,1517..2027) cdslen=1227
Contig1120 40 kDa putative membrane-spanning ATPase
------------------------------------
pans_AUG:pans_0-g1.1 pans_0:join(684..752,818..1464,1517..2027) cdslen=1227
Contig1407 Probable 26S protease subunit and member of the CDC48/PAS1/SEC18 family of ATPases; Rpt5p
------------------------------------
pans_AUG:pans_0-g1.1 pans_0:join(684..752,818..1464,1517..2027) cdslen=1227
PBDEX-M1-006t_A08 ATPase, NSFA, protein involved in protein transport between endoplasmic reticulum and Golg
------------------------------------
```

Figura 3.12: Trecho do arquivo de *matches* entre genes do organismo *Podospora anserina* (Pa) e ESTs do organismo *Paracoccidioides brasiliensis* (Pb).

```
##############################
AO genes with no hits in PB
##############################

================================================================================================
 Identifier
================================================================================================
AO090001000007 AP007154:join(10406..11844,12269..12296) cdslen=1467
AO090001000009 AP007154:complement(join(25153..25587,16084..25080)) cdslen=9432
AO090001000010 AP007154:join(26017..26072,26127..26407,26490..26695,26742..26871
AO090001000011 AP007154:complement(join(30510..30705,30277..30457,30071..30223))
AO090001000018 AP007154:41615..43270 cdslen=1656
AO090001000023 AP007154:join(57665..57975,58037..58595) cdslen=870
AO090001000025 AP007154:complement(join(62298..62392,62146..62237,62033..62135,61728..61977)) cdslen=540
AO090001000026 AP007154:complement(join(62904..62975,62523..62849)) cdslen=399
AO090001000034 AP007154:76702..77472 cdslen=771
AO090001000037 AP007154:complement(join(83716..84113,83116..83662)) cdslen=945
AO090001000042 AP007154:join(90812..91034,91086..91245,91297..92182) cdslen=1269
AO090001000044 AP007154:complement(join(99713..100075,98985..99697,98216..98963)) cdslen=1824
AO090001000046 AP007154:complement(join(102947..103337,102575..102895,101804..102513)) cdslen=1422
AO090001000048 AP007154:104000..104836 cdslen=837
AO090001000055 AP007154:complement(join(117051..117101,116618..116989,116394..116429)) cdslen=459
```

Figura 3.13: Trecho do arquivo de genes específicos do organismo *Aspergillus oryzae* (Ao) em relação as ESTs do organismo *Paracoccidioides brasiliensis* (Pb).

Além do desenvolvimento da ferramenta EGG-LITE, descrevemos como uma nova funcionalidade a implementação da metodologia descrita na Seção 3.1.2 para encontrar as REs, a descrição e implementação de uma nova proposta de metodologia para encontrar as famílias de genes parálogos de um organismo. No Capítulo 4 apresentaremos essa metodologia e sua implementação.

3.4 Outras ferramentas

Descreveremos nessa seção alguns trabalhos que utilizam regiões ortólogas ou estruturas semelhantes para possibilitar a determinação funcional dos genes e a caracterização de aspectos ligados à funcionalidade dos genomas.

Salzberg e colegas [31, 32] descrevem um método para alinhar genomas relacionados distantemente pela detecção de homologia entre seqüências de proteínas. O método, denominado de PROMER, é uma extensão do programa MUMMER [31, 32], e em resumo, alinha dois genomas depois de traduzir suas seqüências de entrada, para cada genoma, em todos os seis *frames de leitura*, extraindo e agrupando todas as seqüências de proteínas que se "relacionaram". Denominam-se esses "relacionamentos" de *matches*.

Especificamente, dado dois arquivos multi-fasta com seqüências de nucleotídeos, PROMER traduzirá essas seqüências em seqüências de aminoácidos considerando os seis *frames de leitura*. Nesse passo, um índice é criado para mapear todas as seqüências de proteínas e seus tamanhos para a seqüência de DNA de origem, sendo este índice usado posteriormente para mapear as seqüências que fizeram *matches* no DNA de origem. As seqüências de aminoácidos transcritas são então filtradas para remover aquelas que tiveram um número excessivo de *stop códons*, indicando que provavelmente não serão partes de uma proteína. Após esses passos, as seqüências transcritas para aminoácidos de cada genoma de entrada

são concatenadas para formarem uma única seqüência de aminoácidos, que representa todas as potenciais proteínas no genoma.

Esses pseudo-proteomas são parâmetros de entrada para o programa MUMMER, que determina todos os *matches* exatos e não exatos com um tamanho maior ou igual a um valor l. O índice criado anteriormente é utilizado para mapear as seqüências *matches* de volta na seqüência de DNA original. Após serem identificados, os *matches* são então agrupados de acordo com suas respectivas coordenadas de DNA. As seqüências *matches* são unidas em um mesmo grupo se o tamanho do intervalo entre elas for menor do que um valor g, determinado pelo usuário. Cada grupo é analisado quanto ao número mínimo de *matches* consecutivos dentro do grupo. Quando esse número de *matches* é no mínimo c, esse grupo será levado em consideração. Os grupos resultantes são então estendidos para "cobrir" uma parte maior da região de alinhamento. Esse passo é realizado utilizando-se o algoritmo de programação dinâmica com "bandas" [36]. As informações do alinhamento resultante são então utilizadas pelo programa "MapView" [32] que mostra os alinhamentos graficamente.

Comparando a metodologia de PROMER com a nossa metodologia de EGG, temos que ambas utilizam grupos de seqüências *matches* para estabelecer regiões de alinhamento entre os genomas de dois organismos. Enquanto PROMER utiliza MUMMER para determinar todas as seqüências *matches* exatas e não exatas, EGG utiliza BLAST para determinar as seqüências que fizeram *hits*, *matches* e BBHs. A metodologia de agrupamento utilizada em EGG é semelhante a utilizada por PROMER, porém EGG utiliza dois parâmetros de entrada, *max_small_gaps* e *max_large_gaps*, para determinar a proximidade entre os RUNs, com pelo menos M *matches*, que farão parte das regiões ortólogas (ROs). Por fim, EGG determina o alinhamento entre os proteomas utilizando o algoritmo de programação dinâmica conforme descrito por Cormen [14], enquanto que PROMER utiliza a versão com "bandas" desse algoritmo. Ambos possibilitam a visualização do alinhamento entre os proteomas, gerando um arquivo de formato "pdf".

Além de possibilitar a visualização do alinhamento entre os proteomas, EGG determina as seqüências e as regiões exclusivas a cada genoma, e mostra essas informações em arquivos textos. EGG gera também arquivos textos para cada produto obtido durante as comparações entre as seqüências, como: os *hits*, *matches*, BBHs, RUNs, ROs, etc.

Enquanto PROMER é utilizado para determinar os alinhamentos entre os genomas baseado na tradução das seqüências nos seis *frames* de leitura, EGG além de possibilitar também essa forma de comparação, permite a comparação das seqüências de DNA dos genomas, sem diretamente traduzí-las. Dessa forma, EGG possibilita "agregar" as funcionalidades dos programas NUCMER e PROMER [31, 32], pacotes de extensão do MUMMER [32], utilizando uma metodologia diferente para detectar as similaridades entre as seqüências em comparação.

Tamames [39] apresentou uma análise da extensão e das características da ordem de conservação dos genes em procariotos, tentando determinar se essa ordem ocorre similarmente para os procariotos e se essas regiões conservadas estão distribuídas uniformemente nos genomas.

Para determinar essas informações, Tamames empregou uma metodologia também baseada na similaridade de BLAST. Nesse caso, para que duas ORFs de genomas distintos

sejam consideradas homólogas, o alinhamento entre elas deve cobrir 75% do tamanho das ORFs e o valor de *e-value* deve ser menor ou igual a 10^{-5}. O relacionamento entre as ORFs que atenderam a essa característica foi denominado de *hits* bidirecionais (BHs). Em EGG, utilizamos o termo "*match*" para definir essa forma de relacionamento. Como genes parálogos podem existir, Tamames utilizou BBHs para determinar os resultados, e utilizou a estrutura dos RUNs para definir um agrupamento de genes no qual a ordem dos genes está conservada. Tamames define o conceito de BBHs como em 3.2.1. Os RUNs que foram utilizados não continham genes pertencentes à fitas diferentes, o que difere de EGG que permite a construção de RUNs com genes de fitas diferentes. Os RUNs de Tamames são construídos com o mínimo de 3 BBHs, permitindo uma distância máxima de 3 genes entre os BBHs. EGG também especifica um parâmetro para o número mínimo de *matches* pertencentes aos RUNs. Para medir a ordem de conservação de genes entre dois genomas, Tamames utilizou a taxa entre o número de genes localizados em RUNs conservados, e o número total de genes que fizeram BBHs. Por fim, métodos de filogenia molecular foram utilizados para determinar o grau de relacionamento entre os organismos [39].

EGG não implementa uma medida de ordem de conservação de genes, no entanto, EGG poderia ser utilizado para determinar a conservação da ordem dos genes entre dois organismos, pois implementa as estruturas necessárias a essa análise. Dessa forma, a conservação da ordem dos genes poderia ser utilizada como uma medida filogenética para estudar os relacionamentos entre as espécies, mesmo entre espécies distantes filogeneticamente, no qual a ordem de conservação dos genes existe na forma de agrupamentos de genes altamente conservados, sugerindo a existência de um processo seletivo que mantém a organização dessas regiões [39]. EGG foi utilizado com esse propósito no trabalho de Araújo [10].

Kellis e colegas [23] apresentaram um método para a determinação automática da correspondência genômica. O método apresentado foi utilizado para o alinhamento entre os genomas das espécies das *Saccharomyces*: *S. paradoxus*, *S.mikatae*, *S.bayanus* e *S.cerevisiae*, permitindo uma correta identificação de genes ortólogos não-ambíguos, cerca de mais de 90% dos não-ambíguos genes codantes dessas proteínas.

Em resumo, o algoritmo representa as similaridades entre os genes como um grafo bipartido, com "peso" nas arestas, conectando genes entre duas espécies. Para "pesar" ou pontuar as arestas que conectam dois genes, foram utilizados ambos, a similaridade entre os aminoácidos das seqüências entre dois genes e o tamanho total de alinhamento entre os genes.

O grafo é separado progressivamente em sub-grafos menores, até que somente os *matches* remanescentes conectem ortólogos "verdadeiros". Essa separação é atingida eliminando arestas que são sub-ótimas em uma série de passos.

No primeiro passo, denominado de pré-processamento, são eliminadas todas as arestas cuja pontuações são menores do que 80% da aresta de máxima pontuação do nó, considerando ambos, a identidade em aminoácidos e o tamanho de alinhamento entre os genes. Como segundo passo, baseado nos *matches* (arestas) não-ambíguos que resultaram do passo anterior, são construídos blocos de genes de ordem conservada (blocos sintênicos) utilizando os *matches* na forma um-para-um entre os genes de espécies distintas. Esses blocos são utilizados para resolver ambigüidades adicionais, entre os genes pertencentes a esses blocos, e genes que não pertencem aos blocos. Finalmente, como terceiro passo,

são procurados os subconjuntos de genes que são ótimos locais, tais que todos os melhores *matches* dos genes dentro de um grupo estão contidos dentro desse grupo; e nenhum gene fora do grupo possui *matches* dentro desse grupo. Esses conjuntos de *matches* são denominados de **BUS** do têrmo em inglês *Best Unambiguous Subsets* e asseguram que o grafo bipartido é maximalmente separável, mantendo todos os possíveis relacionamentos ortólogos [22].

Quando nenhuma separação posterior for possível, as componentes conexas do grafo final são retornadas. Essas componentes contêm pares de genes ortólogos na forma um-para-um.

A metodologia descrita por Kellis [23] difere em alguns aspectos da metodologia empregada em EGG. Em EGG, utilizamos os BBHs, ao invés do BUS. Nesse caso, temos um relacionamento na forma um-para-um entre as seqüências de um genoma contra as seqüências de um outro genoma, sem utilizar diretamente o valor de identidade entre as bases de seqüências comparadas. Utilizamos os valores *e-value* e porcentagem de sobreposição para relacionar dois genes de genomas distintos, formando um BBH da forma descrita na Seção 3.2.1. Assim como Kellis, primeiramente construímos blocos de genes de ordem conservada (RUNs), e em seguida, aumentamos esses blocos para formarem uma região ortóloga, baseado nas informações sintênicas desses genes. Por fim, utilizamos esses BBHs, juntamente com as ROs, para determinar o alinhamento (Espinha Dorsal) entre os conjuntos de proteínas dos genomas.

Embora as metodologias aplicadas em EGG e Kellis [22, 23] considerem medidas diferentes para determinar uma correspondência entre os genes de dois genomas, faz-se necessária uma análise mais refinada entre essas metodologias, utilizando o mesmo conjunto de dados de Kellis [23], para comprovar a afirmação de que a utilização dos BBHs, ao invés do BUS, no caso de um recente evento de duplicação, marca somente um dos genes duplicados como ortólogo, sem sinalizar a presença de homólogos adicionais. Logo, segundo Kellis, o método de EGG estaria apontando relações de ortologia incorretamente, e nesse caso, os BBHs deixariam de representar relacionamentos ortólogos, possibilitando que *matches* sejam determinados incorretamente na forma um-para-um.

Capítulo 4

Determinação de genes parálogos

Devido às duplicações internas que podem ocorrer em um genoma, segundo Cannon e Young [13], para estendermos o conhecimento sobre genes entre espécies filogeneticamente relacionadas, é importante distinguirmos genes que estão diretamente relacionados uns aos outros através de eventos de especiação (ortólogos), dos genes que tiveram duplicação independente da especiação (parálogos).

Como um dos objetivos é determinarmos as famílias de genes parálogos, e como abordamos a determinação dos genes ortólogos na Seção 3.2.1, no presente capítulo, descreveremos a metodologia utilizada por Almeida [1] para obter as famílias de genes parálogos, uma nova proposta para reformular a metodologia de Almeida e a implementação dessa nova proposta.

O capítulo está organizado da seguinte forma. Na Seção 4.1 descrevemos a metodologia aplicada por Almeida [1] e uma nova proposta reformulativa para essa metodologia, e na Seção 4.2 apresentamos os resultados.

4.1 Metodologia

A metodologia de Almeida [1] para encontrar famílias de genes parálogos é baseada nos valores da significância estatística de similaridade e na cobertura de alinhamento, ambos valores disponibilizados por BLAST, que são critérios semelhantes aos utilizados para a determinação dos genes ortólogos, como definido na Seção 3.1.1.

Intuitivamente, a metodologia de Almeida possui as três fases seguintes:

1. Encontrar os genes que são similares entre si e construir famílias com esses genes, considerando valores de significância estatística de similaridade e cobertura de alinhamento entre os genes;

2. Agregar genes externos a essas famílias, segundo valores menos exigentes de significância estatística de similaridade e cobertura de alinhamento; e

3. Para todos os genes que pertencem a mais de uma família, removê-los das famílias com que menos se identificam, segundo o critério de significância de similaridade média entre esses genes e todos os outros genes da família.

Os valores utilizados por Almeida para implementar a significância estatística de similaridade e cobertura de alinhamento foram o *e-value* e a porcentagem média de cobertura entre os genes comparados.

Objetivando melhorias na técnica descrita acima, desenvolvemos juntamente com Almeida e colegas [5], uma nova metodologia para tentar determinar as famílias de genes parálogos, que é baseada em cliques maximais em grafos e modelos ocultos de Markov.

A nova proposta de metodologia consiste de duas fases. Na primeira fase, denominada de "Agrupamento de seqüências", juntamos todas as proteínas em famílias iniciais, baseadas em cliques maximais em grafos. A segunda fase, intitulada "Busca homólogos", constrói um perfil dessas famílias iniciais, segundo os modelos ocultos de Markov. Nas seções seguintes descreveremos estas duas fases.

4.1.1 Agrupamento de seqüências

Essa fase consiste em três passos. No primeiro passo, realizamos a comparação das proteínas na forma todas-contra-todas, utilizando o BLAST como ferramenta comparativa. Consideramos duas proteínas g, h similares se, e somente se, g e h possuírem valores de *e-value* recíprocos menores ou igual a 10^{-5} e com pelo menos 60% de alinhamento entre as seqüências. Utilizamos a condição de porcentagem de cobertura das seqüências no alinhamento com a finalidade de evitarmos o agrupamento das proteínas que compartilham pequenas faixas de alinhamentos, chamadas de "domínios promíscuos" [19]. Proteínas que compartilham somente uma parte pequena desses domínios não compartilham uma história evolucionária e não podem ser membros da mesma família.

O segundo passo consiste em construir um grafo G, onde os vértices são as proteínas e as arestas representam os pares de proteínas similares. A idéia consiste em encontrar estruturas nesse grafo, de tal forma que representem as famílias de proteínas similares. Utilizamos nesse sentido, as cliques em grafos, definidas na Seção 2.2. Especificamente, estamos interessados em cliques maximais. O problema de determinar todas as cliques maximais é NP-Difícil [14], porém na prática, os grafos que representam os relacionamentos de similaridade entre os genes de dois genomas, em grande maioria são esparsos. Além disso, primeiramente separamos o grafo G em componentes conexas e aplicamos o algoritmo de força bruta para encontrar todas as cliques maximais em cada componente, que na maioria são pequenas, em relação ao número de vértices. Ao final desse passo temos todas as cliques maximais de G.

O terceiro passo da primeira etapa consiste na junção de cliques maximais de uma mesma componente que compartilham muitos vértices. Decidimos juntar quaisquer cliques maximais, pertencentes a uma mesma componente, que compartilham pelo menos 50% do número de vértices da menor clique. Denominamos a estrutura resultante de **JMC**, do termo em inglês *Joined Maximal Clique*. Dessa forma, podemos definir uma JMC em G da seguinte forma:

- Uma JMC é uma clique maximal em G; ou
- é a união de duas ou mais JMCs.

Na figura abaixo temos uma representação de duas cliques maximais abstratas C_1 e C_2 formando uma única JMC. A figura ilustra a condição de junção, onde mais do que 50% do número de vértices da clique C_2 são adjacentes com os vértices da clique C_1.

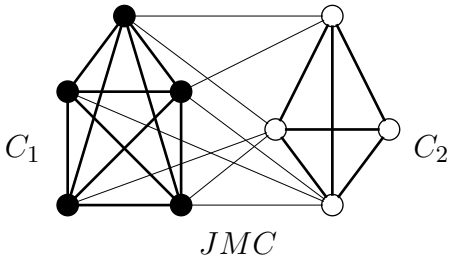


Figura 4.1: Representação de uma JMC. Os vértices de cor preta pertencem a clique C_1, enquanto que os vértices de cor branca pertencem a clique C_2.

O resultado dessa fase é um conjunto de JMCs representando famílias iniciais de proteínas similares. Essas famílias constituem a entrada para a segunda fase, descrita em seguida.

4.1.2 Busca homólogos

A idéia dessa fase consiste em construir modelos para cada JMC, criada na fase anterior, e buscar por proteínas que não fazem parte das JMCs, para que possam ser unidas a essas famílias, segundo um critério de similaridade entre a nova proteína e o modelo.

Nossa metodologia utiliza os modelos estatísticos de alinhamentos múltiplos de seqüência, denominados "Profile Hidden Markov models" (pHMMs) [33]. Esses modelos são utilizados pois retornam informações de posições específicas e a quantidade de conservação de cada coluna do alinhamento, incluindo informações sobre as "penalidades para buracos" de cada posição do alinhamento. Para essa proposta, utilizamos a implementação de pHMMs para a análise de seqüências biológicas, denominada HMMER [17].

Para construirmos os modelos, primeiramente descartamos as seqüências de uma família que apresentam tamanhos discrepantes em relação às outras proteínas da família. Para realizar esse procedimento, excluímos as proteínas de uma determinada família cujo tamanho, em número de aminoácidos, está a uma distância de $D = 3$ desvios padrões do tamanho médio das seqüências da família. O valor D é um parâmetro de entrada para o programa. Em seguida, construímos um alinhamento múltiplo com as seqüências restantes nas famílias, utilizando o programa CLUSTALW [42]. A eliminação das seqüências descrita anteriormente é realizada com a finalidade de aumentar a qualidade do alinhamento múltiplo. Por fim, utilizamos os três programas seguintes abaixo, do pacote HMMER, para construir os modelos e buscar por outras proteínas homólogas.

`hmmbuild` - cria um modelo pHMM para cada família, utilizando o alinhamento múltiplo gerado anteriormente;

`hmmcalibrate` - calibra o pHMM com a finalidade de aumentar a sensibilidade de buscas futuras; e

`hmmsearch` - busca por novos membros para cada família.

Após executar o programa `hmmsearch`, se uma proteína p cujo vértice está em uma mesma componente conexa de uma família F e se a proteína p puder ser agregada à família F, um novo modelo pHMM é construído (incluindo a nova proteína p) e é calibrado novamente para novas buscas. Esse passo é executado até que nenhum pHMM possa ser aumentado. A figura abaixo ilustra o fluxo de execuções da segunda fase da metodologia para uma família abstrata 'A'.

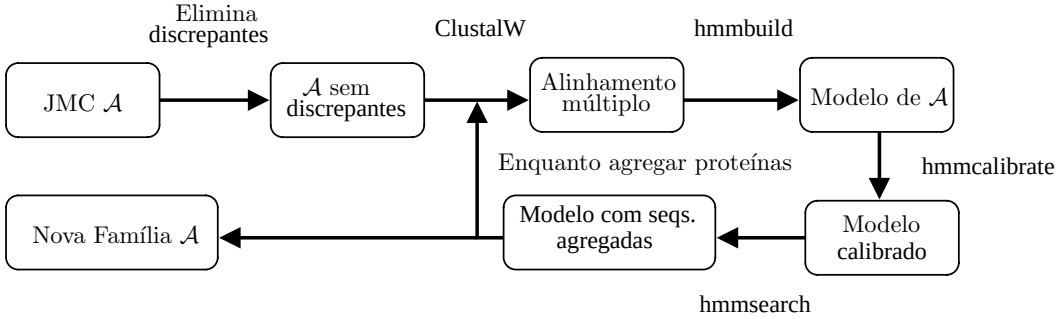


Figura 4.2: Fluxo de execução da fase Busca Homólogos

O resultado dessa fase consiste de famílias com pelo menos três proteínas contendo informações sobre a família, como o identificador, tamanho e produto das proteínas, assim como o alinhamento múltiplo resultante. A figura abaixo é um trecho de um arquivo texto, contendo uma família, resultante da aplicação da nossa metodologia para o organismo *Xylella fastidiosa 9a5c*.

```
  9105737     1032    transcriptional regulator (LysR family)
  9106094      906    transcriptional regulator (LysR family)
  9106570      978    oxidative stress transcriptional regulator
  9106798      891    transcriptional regulator (LysR family)
  9106824      975    transcriptional regulator (LysR family)
  9106842      942    transcriptional regulator (LysR family)
# 9106785      666    hypothetical protein


CLUSTAL W (1.83) multiple sequence alignment


gi|9106824      -----------------MARRNLNDLLYFVTIAREGS-FTRAAAHLGVTQ
gi|9106842      MAEKIVKSTYNGAENSPMPKENLNDLQAFVAVARARS-FTRAAAQLGLSR
gi|9106094      -----------------MQNMFDGVQLFVEVVEAGG-FAKAGKRLSLTR
gi|9106798      ----------------------------MQVVHLGS-FAAAAREQNVDP
gi|9105737      ----MEVIYSRFWFCDISPVMTLTQLRYLVAIADADLNITLAAARIHATQ
gi|9106570      ------MWNYIPSLAARFGFMNLRDLKYLIALADYKH-FGRAATACFVSQ
gi|9106785      --------------------------------------------------


gi|9106824      SALSQAINGLEARLQIRLLTRTTR-SVSPTAAGERLLNAIGHRFDEIESE
gi|9106842      SALSHAMLALEARLGVRLLTRTTR-SVSTTEAGARLLDAVAPRLDEIELE
gi|9106094      STIGKAIARLEMRLGVQLFQRTTR-VQSLTEDGQQYYERCLRAIKELRAG
gi|9106798      SSVSRAVAALEAELGTRLFARNTR-HLALTEAGSVFTERLPLLLEELSQA
gi|9105737      PGLSKQLKKLEDELGFLLFVRRGRSLESVTPAGEEVIERARAMLVEVNNI
gi|9106570      PTLSTQIKKLEGELGVSLVERAPR-KVMMTPAGREAAIRARSIVAEVEEM
gi|9106785      -------------MGALMHDAASR--------------------------
```

Figura 4.3: Uma família encontrada em *Xylella fastidiosa 9a5c*. O símbolo `#` refere-se a uma proteína agregada na segunda fase.

Nossa metodologia foi implementada utilizando scripts e programas escritos em `C++`. O Apêndice B contém maiores detalhes operacionais desses scripts e programas. Na seção seguinte abordaremos alguns resultados da utilização de nossa metodologia, apresentados em [5].

4.2 Resultados

A metodologia foi empregada para encontrar famílias de genes parálogos em cinco genomas. Especificamente, utilizamos os genomas da *Pseudomonas aeruginosa* PA01 (PS), *Escherichia coli* K12 (EC), *Synechocystis sp.* PCC 6803 (SP), *Xylella fastidiosa* 9a5c (XF), e *Chlamydia pneumoniae* (CP). A Tabela 4.1 contém informações encontradas sobre cada família nos genomas.

organismo	N_p	N_{cfc}	N_{fde}	N_{pi}	S_f
PA	5565	16	16	142	0.86
EC	4289	174	170	525	0.81
SP	3169	107	107	264	0.81
XF	2766	57	54	5	0.88
CP	1052	19	18	77	0.84

Tabela 4.1: Informações das famílias encontradas em cada genoma. N_p: número de proteínas; N_{cfc}: número de componentes fortemente conexas com ≥ 3 vértices; N_{fde}: número de famílias depois de descartar seqüências espúrias; N_{pi}: número total de proteínas incluídas; e S_f: Pontuação média das famílias considerando todas as famílias.

Para avaliar nossa metodologia, utilizamos o número COG [40] de cada proteína, obtido no NCBI [49], que supostamente possui uma verificação manual da informações nas seqüências. As proteínas contidas em cada grupo de números COG supostamente possuem alguma relação com alguma proteína ancestral. Para avaliarmos as famílias de proteínas encontradas em cada genoma, utilizamos o valor de *Information Content* (IC), que é uma medida de como é restrita a escolha dos números COG em cada família. O cálculo do IC foi realizado como em [38]. A pontuação final s_f de uma família f é dado pelo seu IC dividido pelo máximo IC que seria possível para a família f. Dessa forma, famílias ideais deveriam possuir $s_f = 1$ onde, $0 \leq s_f \leq 1$. A tabela 4.1 mostra que nossa metodologia, em média, encontrou acima de 80% de parálogos corretos.

Capítulo 5

Comparação de três genomas

Neste capítulo, descrevemos uma proposta [41] para a comparação simultânea de três genomas. A comparação dessas informações é importante pois pode gerar "pistas" das vias metabólicas dos genomas e como as proteínas compartilhadas pelos genomas estão envolvidas em determinadas funções.

Na literatura encontram-se diversas metodologias de comparação genômica [1, 16], muitas delas utilizando informações apenas de seqüências de DNA e outras utilizando seqüências de genes codantes. Dentro desse conjunto de metodologias, existem aquelas que realizam análise comparativa entre genomas de organismos não-patógenos com organismos patógenos para tentar identificar os genes ou grupos de genes associados à doenças e infecções [23].

Em [41] propomos uma metodologia para comparar três genomas simultaneamente, ao nível de suas proteínas preditas. As informações de entrada para a metodologia são os três conjuntos de seqüências de genes codantes, referentes aos três genomas. A saída consiste nas seqüências exclusivas para cada genoma, nas seqüências compartilhadas pelos pares de genomas e nas seqüências compartilhadas pelos três genomas. Comparativamente, a saída da metodologia corresponde às regiões em um diagrama de Venn (Figura 5.1). Determinar cada região do diagrama pode indicar o conjunto de genes compartilhados e exclusivos dos genomas, fornecendo pistas interessantes sobre as vias metabólicas compartilhadas, e também sobre as proteínas relacionadas à uma função em particular.

O método foi utilizado para a comparação de dois genomas de fungos patógenos com cinco genomas de fungos não-patógenos, de três em três simultaneamente, com a finalidade de inferir os genes envolvidos com patogenicidade.

O capítulo está organizado da seguinte forma. Na Seção 5.1 abordamos aspectos gerais da metodologia. Nas seções 5.2 e 5.3 descrevemos aspectos específicos da metodologia e os experimentos. Por fim, na Seção 5.4, apresentamos alguns resultados obtidos.

5.1 Comparando três genomas

Sejam três genomas P, Q, R. Considere que os genomas são representados pelos conjuntos de suas seqüências codantes de DNA ou por suas seqüências de proteínas. Conseguindo atribuir de forma não ambígua os genes de cada genoma à cada região do diagrama de *Venn*, como da Figura 5.1, essas regiões poderiam representar as seqüências compartilhadas e exclusivas de cada genoma.

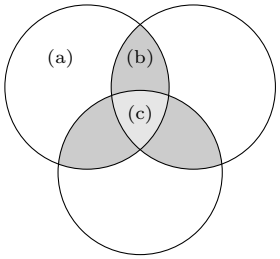


Figura 5.1: Diagrama de Venn representando (a) seqüências exclusivas a um genoma, (b) seqüências compartilhadas por dois genomas, e (c) seqüência compartilhadas pelos três genomas.

A construção do diagrama é realizada utilizando a similaridade de seqüências, com a finalidade de selecionar as seqüências que serão atribuídas a cada região do diagrama. Na Figura 5.2 estão ilustradas algumas situações adequadas de atribuição das seqüências às regiões do diagrama de Venn. As arestas que conectam dois genes indicam que as seqüências desses genes são similares.

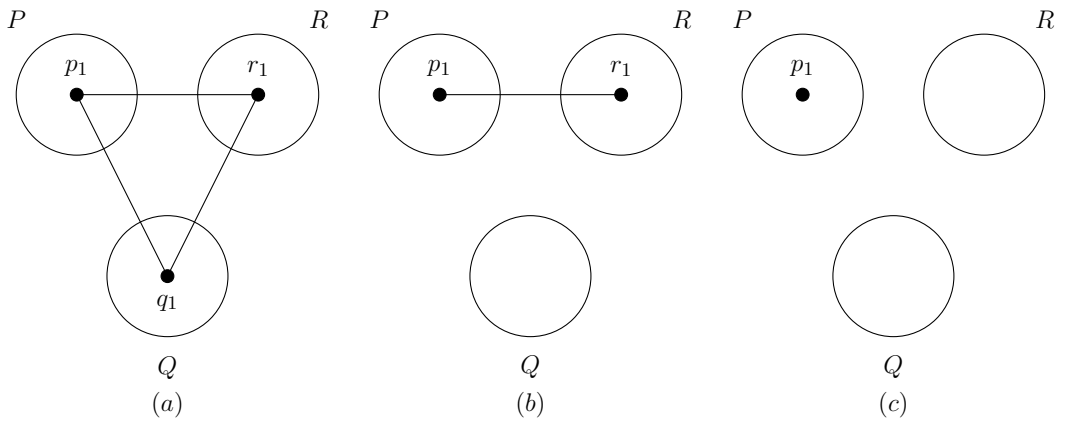


Figura 5.2: Genes que podem ser atribuídos as regiões do diagrama de Venn. (a) Genes em todos os três genomas. (b) Genes pertencentes somente aos genomas P e R. (c) Genes exclusivos a P.

Como uma seqüência de um genoma pode ser similar a muitas outras seqüências de outros genomas, o procedimento de atribuição das seqüências às regiões do diagrama pode

encontrar alguns casos complexos, ou seja, casos onde é difícil decidir em qual região uma seqüência deve ser incluída. Esse processo pode conduzir à relações entre as seqüências com significado biológico não muito evidente. Na Figura 5.3 temos três exemplos dessas relações.

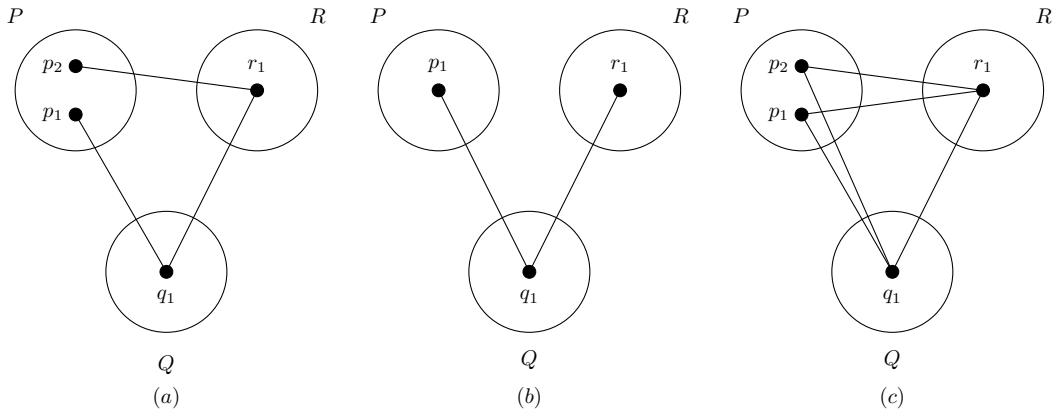


Figura 5.3: Casos complexos de atribuição dos genes às regiões do diagrama de Venn.

Nossa metodologia para encontrar as seqüências em cada região do diagrama de Venn objetiva evitar os casos complexos. O método inicia encontrando as seqüências similares entre três genomas. A busca por essas seqüências similares continua até quando não encontrar mais seqüências compartilhadas entre os três genomas considerando uma determinada pontuação limite T_t para toda tripla de seqüências. Em seguida, buscam-se pelas seqüências comuns a pares de genomas. Esse procedimento continua até quando não encontrar mais seqüências compartilhadas entre dois genomas considerando uma determinada pontuação limite T_e para toda dupla de seqüências. Por fim, as seqüências exclusivas a cada genoma são determinadas. Na seção seguinte descreveremos mais detalhadamente esse método.

5.2 Descrição do Método

As seguintes considerações são necessárias para descrevermos detalhadamente o método :

- Uma seqüência é apenas uma justaposição de letras;
- As seqüências podem ser de DNA ou proteína;
- Um genoma consiste de um conjunto de seqüências; e
- Denota-se $G(p)$ o genoma ao qual a seqüência p pertence.

O método utiliza os seguintes conceitos de aresta e triângulo.

Definição 5.1 (Aresta entre genomas) *Dados dois genomas P e Q, uma* ***aresta*** *é um par de seqüências (p, q), $p \in P$ e $q \in Q$. O peso w de uma aresta (p, q), denotado por $w(p, q)$, é qualquer medida de similaridade que possa avaliar p e q.*

Definição 5.2 (Triângulo entre genomas) *Dados três genomas P, Q e R, um* ***triângulo*** *é uma tripla de seqüências (p, q, r), $p \in P$, $q \in Q$ e $r \in R$, tais que existam as arestas (p, q), (p, r) e (q, r). O peso w de um triângulo (p, q, r), denotado por $w(p, q, r)$, é qualquer medida que possa avaliar $w(p, q)$, $w(p, r)$ e $w(q, r)$.*

O método começa calculando o peso de cada par de seqüências de genomas distintos. Em seguida, os triângulos são processados em ordem não-crescente dos seus pesos. Um a um os triângulos são atribuídos às regiões do diagrama de Venn, até que não existam mais triângulos com peso maior ou igual a T_t. Quando um triângulo (p, q, r) é atribuído à região central do diagrama, outros triângulos ou arestas que contêm os vértices p, q ou r não são levados mais em consideração. Em seguida, as arestas são processadas uma a uma em ordem não-crescente dos seus pesos. Uma após outra as arestas são atribuídas às regiões de intersecção entre dois genomas do diagrama, até que não existam mais arestas com peso maior ou igual a T_e. Quando uma aresta (u, v) é atribuída à uma região do diagrama, outras arestas ou vértices que contêm u ou v não são mais levados em consideração. Na página seguinte descrevemos em pseudo-código o algoritmo intitulado MÉTODO-3GC que foi desenvolvido como a descrição dessa metodologia.

A nossa metodologia é uma heurística gulosa, no sentido de que para cada triângulo e aresta selecionados, sempre optamos por atribuir primeiramente aqueles que possuem os maiores pesos, sem garantia de que esses triângulos e arestas evidenciem relações biologicamente corretas entre os genes desses organizas.

O tempo de execução para um algoritmo que segue os passos da metodologia é da ordem de $O(|P||Q|\alpha + |P||R|\alpha + |Q||R|\alpha + |P||Q||R|\beta + |P||Q|R| \log(|P||Q|R|))$, que corresponde ao número de comparações de seqüências à um custo α por comparação, mais a quantidade de triângulos que podem ser formados a um custo β por triângulo. A última parcela da soma corresponde ao custo da ordenação dos triângulos e arestas.

Algoritmo 4 MÉTODO-3GC

Entrada: Genomas P, Q e R

Saída: seqüências em P, Q e R atribuídas às regiões do diagrama de Venn.

1: $L \leftarrow \emptyset$.
2: Calcule $w(a, b)$ para todo par de seqüências tal que $G(a) \neq G(b)$.
3: **para** todo triângulo (p, q, r), $p \in P$, $q \in Q$ e $r \in R$, tal que $w(p, q, r) \geq T_t$ **faça**
4: acrescente (p,q,r) à lista L;
5: **fim para**
6: Ordene a lista L na forma não-crescente em relação aos pesos dos triângulos.
7: **enquanto** $L \neq \emptyset$ **faça**
8: pegue o primeiro triângulo de L e renomeie para t, onde $t = (p, q, r)$.
9: acrescente t à região do diagrama.
10: remova qualquer triângulo em L que possua p, q ou r como membro
11: remova p, q, r dos genomas P, Q e R.
12: **fim enquanto**
13: **para** toda aresta (p, q), $p \in P$ e $q \in Q$ tal que $w(p, q) \geq T_e$ **faça**
14: acrescente (p, q) à lista L;
15: **fim para**
16: **para** toda aresta (p, r), $p \in P$ e $r \in R$ tal que $w(p, r) \geq T_e$ **faça**
17: acrescente (p, r) à lista L;
18: **fim para**
19: **para** toda aresta (q, r), $q \in Q$ e $r \in R$ tal que $w(q, r) \geq T_e$ **faça**
20: acrescente (q, r) à lista L;
21: **fim para**
22: Ordene a lista L na forma não-crescente em relação aos pesos das arestas.
23: **enquanto** $L \neq \emptyset$ **faça**
24: pegue a primeira aresta de L e renomeie para e, onde $e = (a, b)$.
25: acrescente e à região do diagrama.
26: remova qualquer aresta em L que contenha a ou b como membro.
27: remova a e b de $G(a)$ e $G(b)$ respectivamente.
28: **fim enquanto**
29: **para** toda seqüência s em P **faça**
30: acrescente s à região correspondente exclusivamente a P no diagrama.
31: **fim para**
32: **para** toda seqüência s em Q **faça**
33: acrescente s à região correspondente exclusivamente a Q no diagrama.
34: **fim para**
35: **para** toda seqüência s em R **faça**
36: acrescente s à região correspondente exclusivamente a R no diagrama.
37: **fim para**

5.3 Experimentos

Os experimentos foram realizados utilizando genomas de fungos. Especificamente, foram utilizados cinco genomas de fungos não-patógenos e dois genomas de fungos patógenos. Utilizamos os seguintes genomas: *Aspergillus nidulans*(9541 seqüências), *Candida albicans*(6165 seqüências), *Criptococcus neoformans*(6578 seqüências), *Fusarium graminearum*(11640 seqüências), *Magnaporte grisea*(11109 seqüências), *Neurospora crassa*(10082 seqüências) e *Saccharomyces cereviseae*(6305 seqüências). *C. neoformans* e *C. albicans* são fungos patógenos de humanos, enquanto que os outros não são. Nesses experimentos, foi definido que uma aresta entre dois vértices (genes) ocorre através da utilização dos *hits* bidirecionais do programa BLAST. Dizemos que as seqüências p e q, onde $p \in P$ e $q \in Q$ formam um hit bidirecional se:

- q é encontrado por BLAST de p contra Q com **e-value** menor ou igual a 10^{-5}; e
- p é encontrado por BLAST de q contra P com **e-value** menor ou igual a 10^{-5}.

O peso de uma aresta (p, q) é obtido pela cobertura média do alinhamento das seqüências p e q retornada pelo relatório do BLAST. A cobertura média de alinhamento foi utilizada pois acredita-se que essa medida possa evitar que uma aresta entre dois genes compartilhe apenas pequenos "pedaços" das seqüências (domínios espúrios [19]).

Para determinar um valor adequado para T_t e T_e, foi utilizado o banco de dados *Pfam* [11] como referência comparativa. Pfam foi escolhido pois utiliza *Hidden Markov Models* para detectar características de famílias de proteínas de uma dada seqüência proteica. Espera-se que seqüências de um mesmo triângulo pertençam a uma mesma família Pfam.

Os testes realizados em [41] mostraram que o valor 50 é adequado para ambos os limites T_t e T_e, pois foi observado um número constante de triângulos e arestas para uma cobertura de alinhamento abaixo de 50%. A metodologia aparentemente previne que *hits* contendo baixa similaridade façam parte dos triângulos, o que melhora o número de ortólogos detectados corretamente entre as espécies comparadas.

5.4 Alguns resultados

A metodologia permitiu a comparação e extração de informações entre genomas patógenos e não-patógenos. Os triângulos entre os genomas comparados sugerem que esses genes estão envolvidos no metabolismo central dos genomas, considerando esse conjunto de genes indispensáveis para fungos. Outras inferências biológicas podem ser vistas em [41].

Como um outro resultado, essa metodologia surge como um *framework*, no sentido de que alguns valores e limites não foram especificados, permitindo que o método possa ser especializado dependendo da necessidade, tal como o poder de processamento, tamanho do genomas, e sensibilidade.

A comparação de todos os genomas de fungos não-patógenos com os genomas *C.**neoformans*** e *C.**albicans*** gerou os cinco diagramas de *Venn* da Figura 5.4 abaixo.

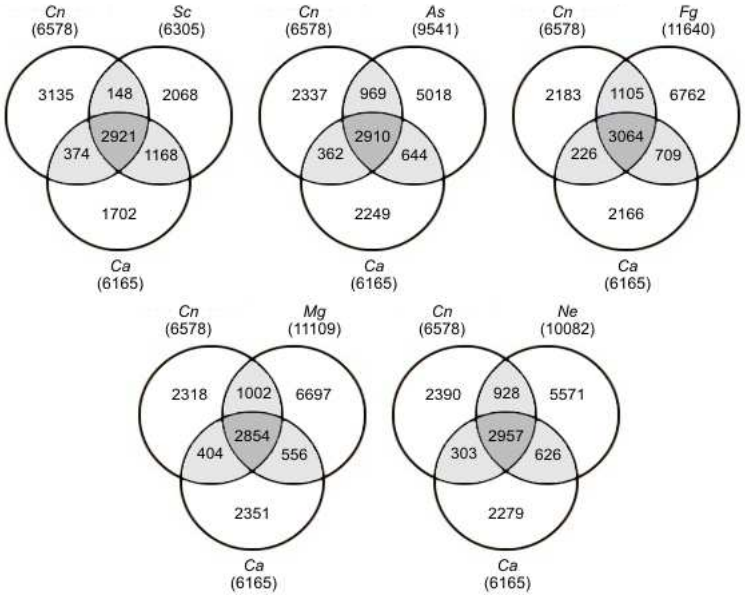


Figura 5.4: Diagramas de Venn gerados pela comparação entre dois genomas de fungos patógenos e cinco genomas de fungos não-patógenos, comparados de três em três simultaneamente.

Para apresentarmos os resultados obtidos, desenvolvemos dois scripts em Perl. O script intitulado VENN.PL gera uma figura de diagrama de Venn preenchida com os valores das intersecções, com os nomes dos organismos em comparação e com o número de genes de cada organismo. O segundo script, VENNWEB.PL, gera uma página HTML com a figura do diagrama de Venn, anteriormente gerada, com links na figura para arquivos textos referentes às regiões do diagrama de Venn que são compartilhadas entre dois genomas, entre os três genomas e para as regiões exclusivas de cada genoma.

Capítulo 6

Considerações finais

O presente trabalho teve como principal objetivo a reformulação do pacote de ferramentas EGG [1]. A reformulação incluiu a reimplementação de todo o código-fonte, além da descrição e implementação de novas metodologias e funcionalidades.

As principais contribuições do trabalho foram:

- total reformulação do código-fonte dos programas contidos na ferramenta EGG;
- implementação de metodologia para encontrar regiões específicas – essa metodologia foi proposta em [1] mas não havia sido implementada;
- desenvolvimento e implementação de nova metodologia para encontrar genes parálogos (genes homólogos pertencentes ao mesmo organismo);
- desenvolvimento e implementação de uma versão de EGG, denominada EGG-LITE, para a comparação de genomas incompletos, onde se tem apenas genes, ou transcritos do organismo, sem o seqüenciamento completo dos cromossomos;
- desenvolvimento e implementação de um módulo para a comparação de três proteomas.

Após essa reformulação, os códigos-fonte de EGG, além de manuais e documentação se encontram disponíveis para *download* em `http://egg.dct.ufms.br/egg`.

A aplicação das novas metodologias inseridas em EGG foram utilizadas em trabalhos de pesquisa e resultaram na co-autoria das publicações [5, 41] e do artigo em preparação [47]. Além disso, a experiência adquirida durante o programa permitiu ainda a co-autoria de [43, 44].

Trabalhos futuros

Muitas alterações podem ser realizadas no presente trabalho para melhorar o pacote de ferramentas EGG, suas metodologias e o detalhamento dos seus resultados. A seguir, listamos algumas das possibilidades.

Interface gráfica

Os arquivos de saída do pacote de programa EGG estão em formato texto, a menos dos arquivos de plotagem `cbbh.pdf` e `hbbh.pdf`. Maiores detalhes desses arquivos estão no apêndice A. Embora os arquivos de saída possam ser processados/analisados, muitas vezes são difíceis de serem amplamente analisados. Dessa forma, uma interface gráfica que centralizasse as informações dos arquivos textos seria muito útil.

Portanto, poderíamos criar uma ferramenta que utilizasse informações de ortologia entre proteínas ilustrando as interações entre as proteínas e entre as regiões em que estão localizadas. A idéia consiste em ilustrar graficamente as regiões ortólogas e as regiões específicas, permitindo a visualização das relações com perspectivas para escalas menores, com *zoom's*, de forma similar a utilizada pela ferramenta STRING [50], utilizando informações originárias dos RUNs, ROs e REs, ao invés das informações de COG, utilizadas pela ferramenta STRING [50].

Sintonia-fina para determinar parálogos

Muitas alterações podem ser feitas em nossa metodologia para determinar genes parálogos. Primeiro, podemos melhorar o algoritmo para encontrar as JMCs, por exemplo, atribuindo pesos ou pontuações às arestas do grafo. Podemos também tentar evitar o caso de que duas ou mais famílias compartilhem uma mesma proteína, decidindo que uma proteína deva participar da família com a qual mais se identifique [1]. Um outro trabalho futuro está relacionado com a qualidade dos alinhamentos. Nesse caso podemos ajustar os parâmetros do programa `hmmbuild` para melhorar a qualidade dos pHMMs resultantes. Precisamos também encontrar valores mais adequados para todos os parâmetros utilizados em nossa metodologia, pois os valores que utilizamos foram baseados em testes preliminares.

Agrupamento na comparação entre 3 genomas

Ao compararmos três genomas, sugerimos como um outro trabalho futuro, a implementação de um passo prévio de *clusterização* (agrupamento) dos três genomas. Nesse caso, esse passo adicional na metodologia poderia ajudar a evitar os casos complexos envolvendo os genes parálogos, descritos na Seção 5.1. Poderíamos também aplicar a metodologia de encontrar regiões específicas, descrita na Seção 3.1.2, aos pares de genomas, para tentar determinar as possíveis regiões específicas em cada genoma, tentando descobrir metabolismos secundários de cada organismo.

Apêndice A

Detalhes operacionais de EGG

A nova proposta de implementação para a metodologia descrita na Seção 3.2 resultou nos programas EGG e EGG-LITE. Nas seções seguintes descreveremos alguns detalhes sobre esses programas.

A.1 Descrição das Ferramentas

Os programas e *scripts* pertencentes aos pacotes de EGG e EGG-LITE foram escritos nas linguagens `C++` e `Perl`. Os programas escritos em `C++` utilizaram o compilador `g++` (GNU Software).

Conforme descrito na Seção 3.2.1, o programa EGG contém três fases (comparação de genes, determinação do grafo bipartido e determinação das estruturas organizacionais). No entanto, a implementação de EGG foi dividida em dois módulos. No primeiro módulo implementamos a comparação entre as seqüências na forma todas-contra-todas, e no segundo módulo implementamos a "construção" do grafo bipartido e a determinação das estruturas organizacionais, como descrito na Seção 3.2.1. Os módulos de EGG são executados separadamente.

O programa EGG-LITE, conforme descrito em 3.3, contém apenas as duas fases iniciais de EGG. A implementação dessas fases foram realizadas em dois módulos, conforme EGG. No entanto, os módulos de EGG-LITE são executados em apenas um passo. Na seções seguintes abordaremos maiores detalhes sobre cada programa.

EGG

A ferramenta EGG possui como entrada dois arquivos com os conjuntos de seqüências dos genes ou proteínas dos genomas G e H. EGG supõe a existência de dois arquivos com extensões `ptt`, que trazem informações sobre os genes dos genomas. As informações contidas nesses arquivos são as coordenadas de início e fim, a fita, tamanho do gene, nome, produto, código identificador no *GenBank* (gi) e número COG.

Além dos arquivos com extensão `ptt`, EGG supõe que existam também dois arquivos com extensões `faa` que contêm as seqüências no formato FASTA, de todos os genes dos

genomas. Ambos os tipos de arquivos estão disponíveis no NCBI [49] para genomas publicados.

Conforme descrito na Seção 3.2.1, EGG compara cada gene de um genoma contra todos os genes do outro genoma, e vice-versa, utilizando a ferramenta BLAST. Essas comparações são realizadas primeiramente construindo as bases de seqüências de genes para cada genoma. Utilizamos o programa `formatdb`, que é uma ferramenta disponível pelo pacote do BLAST, para construir as bases de seqüências. Após construir essas bases para cada genoma, o programa EGG pode ser executado.

O pacote EGG contém dois módulos. O primeiro módulo, denominado de EGG-PARSER, busca informações, geradas pelo BLAST, sobre as comparações entre as seqüências pertencentes aos arquivos de extensões `faa` e gera arquivos de saída contendo informações necessárias para o segundo módulo. Supondo que o nome dos genomas em comparação são `gG` e `gH`, os seguintes arquivos de entrada são necessários:

- `gG.ptt` e `gH.ptt`; e
- `gG.faa` e `gH.faa`.

Os arquivos de saída gerados por EGG-PARSER possuem os seus nomes compostos pelos nomes dos arquivos de entrada de extensão `ptt`. Assim, teríamos os seguintes arquivos de saída e suas descrições:

- `gGgH.12` - Para cada gene do genoma `gG`, contém as informações sobre a quantidade de hits encontrados; e para cada hit, o número seqüencial do hit, o seu gi, score, e-value e as porcentagens de cobertura de alinhamento entre o gene e o hit;
- `gGgH.21` - Para cada gene do genoma `gH`, contém as informações sobre a quantidade de hits encontrados; e para cada hit, o número seqüencial do hit, o seu gi, score, e-value e as porcentagens de cobertura de alinhamento entre o gene e o hit;
- `gGgH.1` - Contém as mesmas informações do arquivo `gG.ptt`, porém em um formato mais facilmente parseável; e
- `gGgH.2` - Contém as mesmas informações do arquivo `gH.ptt`, porém em um formato mais facilmente parseável.

Na Seção A.2 apresentamos alumas figuras ilustrando trechos dos arquivos de saída de EGG-PARSER.

EGG-PARSER contém um conjunto de parâmetros que permitem ao usuário executar e alterar valores de alguns dos parâmtros de entrada do BLAST. A descrição da lista de parâmetros pode ser vista na documentação do programa, encontrada em [48].

O objetivo da criação dos arquivos de saída é permitir que possam ser facilmente utilizados por outros programas que necessitem dessas informações.

A implementação desse módulo foi escrita em `Perl`, devida a flexibilidade da sua linguagem para a extração de informações de documentos textos, utilizando expressões regulares.

O segundo módulo, denominado simplesmente de EGG, conforme descrito em 3.2.1, "constrói" um grafo bipartido e encontra as estruturas organizacionais nesse grafo. EGG utiliza como parâmetros de entrada os arquivos de extensão `ptt` e os arquivos gerados por EGG-PARSER. Supondo novamente que os nomes dos arquivos com extensão `ptt` são `gG` e `gH`, EGG necessita dos seguintes arquivos:

- `gG.ptt` e `gH.ptt`;
- `gGgH.1` e `gGgH.2`; e
- `gGgH.12` e `gGgH.21`

Os nomes dos arquivos de saída são compostos pelos nomes dos arquivos de extensão `ptt`. Assim teríamos os seguintes arquivos de saída e suas funcionalidades:

- `gGgH.bbh` - Informações sobre os genes que fizeram BBH. Esse arquivo foi utilizado em [10] como instrumentos para a obtenção de medidas de distâncias entre genomas em comparação.
- `gGgH.bkb` - Informações sobre a espinha dorsal entre `gG` e `gH`;
- `gHgG.bkb` - Informações sobre a espinha dorsal entre `gH` e `gG`;
- `gGgH.exc` - Informações sobre os genes exclusivos de cada genoma;
- `gGgH.hyp` - Informações sobre os genes com produtos hipotéticos;
- `gGgH.k12` - Informações para cada gene do genoma `gG`, se o gene possui hits, o valor de e-value do melhor hit e em quais regiões o hit aparece;
- `gGgH.k21` - Informações para cada gene do genoma `gH`, se o gene possui hits, o valor de e-value do melhor hit e em quais regiões o hit aparece;
- `gGgH.mul` - Informações simplificadas de cada região ortóloga. Esse arquivo foi utilizado em [25] para a comparação de múltiplos genomas;
- `gGgH.reg` - Informações mais detalhadas de cada região ortóloga;
- `gGgH.rns` - Informações sobre os runs entre os genomas;
- `gGgH.sre` - Informações sobre as regiões específicas do genoma `gG`;
- `gHgG.sre` - Informações sobre as regiões específicas do genoma `gH`;
- `cbbh.pdf` - Gráfico ilustrando os BBHs; e
- `hbbh.pdf` - Gráfico ilustrando as ROs;

O programa EGG contém um conjunto de parâmetros que possibilitam especificar: o número mínimo de ORFs dentro de uma região específica, o número mínimo de *matches* dentro de um *RUN*, os valores de e-value para a definição de um *match*, porcentagem mínima de cobertura de alinhamento entre as seqüências comparadas, tamanho dos genomas, etc. A lista completa dos parâmetros pode ser encontrada no endereço eletrônico: `http://egg.dct.ufms.br/egg/`

EGG-LITE

A ferramenta necessita como parâmetros de entrada os dois arquivos de extensão `faa`, que contém as seqüências no formato FASTA.

Descrevemos na Seção 3.3 que EGG-LITE compara uma seqüência de um conjunto A de seqüências, contra todas as seqüências de um conjunto B e vice-versa. Conforme EGG, EGG-LITE utiliza o BLAST como ferramenta comparativa e o programa `formatdb` para criar as bases de seqüências.

O pacote EGG-LITE é composto por dois módulos. O primeiro módulo, denominado simplesmente de EGG-LITE, é responsável por definir as relações de *match* e BBH entre os conjuntos de seqüências utilizando como parâmetros de entrada os arquivos de saída do segundo módulo. O segundo módulo, denominado de EGG-PARSER-LITE, executa as comparações das seqüências dos conjuntos na forma todas-contra-todas, utilizando o BLAST e criando como saída, arquivos contendo as seqüências de um conjunto e suas seqüências hits do outro conjunto.

Conforme descrito na Seção 3.3, EGG-LITE não constrói as estruturas organizacionais entre os conjuntos de seqüências, pois poderíamos estar comparando um genoma contra um conjunto de seqüências de um outro genoma ainda não completamente seqüenciado (ESTs). Supondo que os nomes dos conjuntos de seqüências são `sG` e `sH`, EGG-LITE gera os seguintes arquivos de saída:

- `sGsH.bbh` - Informações sobre as seqüências que fizeram BBH;
- `sGsH.exc` - Informações sobre as seqüências que são exclusivas em cada conjunto; e
- `sGsH.match` - Informações sobre as seqüências que fizeram *match*;

Assim como em EGG, EGG-LITE possui um conjunto de parâmetros que permitem a execução do programa BLAST e a alteração dos valores de alguns parâmetros do BLAST, para a sua execução. A lista de parâmetros de EGG-LITE pode ser obtida em [48].

O programa EGG-LITE foi escrito em `C++` e o script EGG-PARSER-LITE foi escrito em `Perl`. Algumas figuras contendo trechos dos arquivos de saída de EGG-LITE podem ser vistas em 3.11, 3.12 e 3.13.

Os pacotes EGG, EGG-LITE, e uma documentação dos programas e scripts contidos nesses pacotes podem ser obtidos no endereço eletrônico: `http://egg.dct.ufms.br/egg/`

A.2 Exemplos de alguns arquivos de saída

As figuras A.1 e A.2 ilustram um trecho dos arquivos `xacxcc.12` e `xacxcc.1` resultantes da comparação dos genomas *Xanthomonas axonopodis pv. citri str. 306* (xac) e *Xanthomonas campestris pv. campestris str. ATCC 33913* (xcc).

```
#1   21240775  2
> 1        21229479       860        0.0      100      100
> 2739     21232217        46      4e-06       49       87
#2   21240776  1
> 2        21229480       676        0.0      100      100
#3   21240777  1
> 3        21229481       655        0.0       98       98
#4   21240778  2
> 4        21229482      1545        0.0      100      100
> 1670     21231148       362     1e-101       73       93
#5   21240779  1
> 5        21229483       382     1e-107      100       81
#6   21240780  2
> 6        21229484       509     1e-146      100      100
> 955      21230433        83      2e-17       60       31
#7   21240781  1
> 7        21229485       714        0.0      100      100
```

Figura A.1: Trecho do arquivo `xacxcc.12`. Cada gene do genoma `xac` é marcado com `#`, seguido pelo seu número seqüencial no genoma `xac`, de acordo com seu arquivo de extensão `ptt`, pelo seu gi e pela quantidade de *hits*. Cada hit é marcado com o símbolo `>`, seguido pelo seu número sequencial no genoma `xcc`, pelo seu gi, score, e-value, porcentagens de cobertura de alinhamento entre o gene e o hit.

```
4312
21240775    42          1370        +   dnaA        XAC0001     chromosomal replication initiator
21240776    1647        2747        +   dnaN        XAC0002     DNA polymerase III beta chain
21240777    3799        4905        +   recF        XAC0003     DNA replication and repair RecF protein
21240778    5020        7464        +   gyrB        XAC0004     DNA gyrase subunit B
21240779    7685        8368        +   -           XAC0005     hypothetical protein
21240780    8552        9358        +   -           XAC0006     hypothetical protein
21240781    9636        10829       +   -           XAC0007     hypothetical protein
21240782    10983       11654       +   tonB        XAC0008     TonB protein
21240783    11740       12501       +   exbB        XAC0009     biopolymer transport ExbB protein
21240784    12548       12970       +   exbD1       XAC0010     biopolymer transport ExbD1 protein
21240785    12974       13387       +   exbD2       XAC0011     biopolymer transport ExbD2 protein
21240786    13649       14416       -   pdxJ        XAC0012     pyridoxal phosphate biosynthetic protein
21240787    14424       14756       -   -           XAC0013     hypothetical protein
21240788    14768       16228       -   cls         XAC0014     cardiolipin synthetase
21240789    16671       17330       +   -           XAC0015     hypothetical protein
21240790    17330       17920       +   -           XAC0016     hypothetical protein
21240791    18131       19258       -   -           XAC0017     hypothetical protein
21240792    19442       20359       -   -           XAC0018     hypothetical protein
21240793    20413       21753       -   ompP1       XAC0019     outer membrane protein
```

Figura A.2: Trecho do arquivo `xacxcc.1` . Na primeira linha do arquivo temos a quantidade de orfs encontradas no arquivo `xac.ptt`. Em cada linha subseqüente, temos informações das orfs, como: gi, coordenada de início e fim no genoma, fita, nome, sinônimo, e produto, separados por espaços em branco.

A Figura A.3 mostra um trecho do arquivo de saída de extensão **bbh** resultante da comparação entre os genomas das Alphaproteobacterias *Agrobacterium tumefaciens str. C58* (at) e *Sinorhizobium meliloti 1021* (sm).

```
################################
ATSM - bidirectional best hits
################################

=================================================================================================================
 Gene     Synonym        gi         start..end       size  product
=================================================================================================================
+_        Atu0001    17933926        203..1024      273aa hypothetical protein
+_        SMc02793   15963754        478..1299      273aa hypothetical protein
---------------------------------
+maf      Atu0002    17933927       1056..1655      199aa Maf like protein
+maf      SMc02792   15963755       1347..1946      199aa Maf like protein
---------------------------------
+aroE     Atu0003    17933928       1648..2508      286aa shikimate 5 dehydrogenase
+aroE     SMc02791   15963756       1939..2799      286aa shikimate 5 dehydrogenase
---------------------------------
+coaE     Atu0004    17933929       2505..3089      194aa dephospho CoA kinase
+coaE     SMc02790   15963757       2796..3380      194aa dephospho CoA kinase
---------------------------------
+dnaQ     Atu0005    17933930       3110..3808      232aa DNA polymerase III subunit epsilon
+dnaQ     SMc02789   15963758       3574..4302      242aa DNA polymerase III subunit epsilon
---------------------------------
-secB     Atu0006    17933931       3907..4389      160aa export protein SecB
-secB     SMc02788   15963759       4408..4914      168aa export protein SecB
---------------------------------
-fxsA     Atu0007    17933932       4493..5029      178aa hypothetical protein
-_        SMc02787   15963760       5082..5657      191aa hypothetical protein
---------------------------------
+_        Atu0008    17933933       5215..5868      217aa hypothetical protein
+_        SMc02786   15963761       5792..6493      233aa PUTATIVE TRANSLOCASE TRANSMEMBRANE PROTEIN
---------------------------------
+mltA     Atu0009    17933934       5865..6977      370aa membrane bound lytic murein transglycosylase
+_        SMc02785   15963762       6511..7629      372aa PUTATIVE LYTIC MUREIN TRANSGLYCOSYLASE A PROTEIN
---------------------------------
+_        Atu0010    17933935       6974..7543      189aa hypothetical protein
+_        SMc02784   15963763       7622..8188      188aa hypothetical protein
---------------------------------
```

Figura A.3: Trecho do arquivo `atsm.bbh`. As colunas informam da esquerda para a direita, a fita, o nome do gene; o sinônimo do gene; o identificador no GenBank; as coordenadas de DNA; o tamanho do gene e o nome do produto.

A Figura A.4 mostra um trecho do arquivo `xacxcc.bkb` resultante da comparação entre *Xanthomonas axonopodis pv. citri str. 306* (xac)e *Xanthomonas campestris pv. campestris str. ATCC 33913* (xcc).

```
=============================================================================================================================
                          PRODUCT  START..END    GENE(STRAND)        (STRAND)GENE   START..END    PRODUCT
=============================================================================================================================
  chromosomal replication initiator     42..1370   XAC0001 (+)  <<<>>>  (+) XCC0001      42..1370   chromosomal replication initiator
      DNA polymerase III beta chai   1647..2747   XAC0002 (+)  <<<>>>  (+) XCC0002    1646..2746   DNA polymerase III beta chain
    DNA replication and repair RecF   3799..4905   XAC0003 (+)  <<<>>>  (+) XCC0003    3633..4739   DNA replication and repair RecF protein
                 DNA gyrase subunit   5020..7464   XAC0004 (+)  <<<>>>  (+) XCC0004    4853..7297   DNA gyrase subunit B
              hypothetical protein   7685..8368   XAC0005 (+)  <<<>>>  (+) XCC0005    7359..8201   hypothetical protein
              hypothetical protein   8552..9358   XAC0006 (+)  <<<>>>  (+) XCC0006    8264..9070   hypothetical protein
              hypothetical protein   9636..10829  XAC0007 (+)  <<<>>>  (+) XCC0007    9209..10405  hypothetical protein
                        TonB protein  10983..11654  XAC0008 (+)  <<<>>>  (+) XCC0008   10559..11230  TonB protein
  biopolymer transport ExbB protein  11740..12501  XAC0009 (+)  <<<>>>  (+) XCC0009   11315..12076  biopolymer transport ExbB protein
biopolymer transport ExbD1 protein  12548..12970  XAC0010 (+)  <<<>>>  (+) XCC0010   12123..12545  biopolymer transport ExbD1 protein
biopolymer transport ExbD2 protein  12974..13387  XAC0011 (+)  <<<>>>  (+) XCC0011   12549..12959  biopolymer transport ExbD2 protein
   pyridoxal phosphate biosynthetic  13649..14416  XAC0012 (-)  <<<>>>  (-) XCC0012   14113..14883  pyridoxal phosphate biosyn
              hypothetical protein  14424..14756  XAC0013 (-)  <<<>>>  (-) XCC0013   14891..15160  hypothetical protein
             cardiolipin synthetas  14768..16228  XAC0014 (-)  <<<>>>  (-) XCC0014   15235..16695  cardiolipin synthetase
              hypothetical protein  16671..17330  XAC0015 (+) #            -
              hypothetical protein  17330..17920  XAC0016 (+) #            -
              hypothetical protein  18131..19258  XAC0017 (-)  <<<>>>  (+) XCC0015   16981..18075  hypothetical protein
              hypothetical protein  19442..20359  XAC0018 (-)  <<<>>>  (-) XCC0016   18170..18940  hypothetical protein
            outer_membrane protein  20413..21753  XAC0019 (-)  <<<>>>  (-) XCC0017   19513..20853  outer membrane protein
              hypothetical protein  21972..22664  XAC0020 (+)  <<<>>>  (+) XCC0018   21074..21766  hypothetical protein
```

Figura A.4: Exemplo de uma espinha dorsal entre os genomas dos organismos *Xanthomonas axonopodis pv. citri str. 306* (Xac) e *Xanthomonas campestris pv. campestris str. ATCC 33913* (Xcc). Cada símbolo `<<<>>>` mostra um BBH pertencente a espinha dorsal. O símbolo `#` indica que o gene é específico; o símbolo `*` indica que o gene fez hit, mas não fez BBH. Quando um gene aparece sem "casamento" e sem qualquer símbolo, temos uma translocação, ou seja, ele fez BBH, mas esse BBH não ficou aparente porque está cruzando com outros, e esses outros não foram mostrados pela programação dinâmica.

A Figura A.5 mostra um trecho do arquivo de saída de extensão **exc** resultante da comparação entre os genomas das Alphaproteobacterias *Agrobacterium tumefaciens str. C58* (at) e *Mesorhizobium loti MAFF303099* (ml).

```
##############################
AT genes with no hits in ML
##############################

=============================================================================================
 Gene      Synonym       gi         start..end        size  product
=============================================================================================
-_         Atu0013    17933938      10604..11206     200aa  NAD(P)H flavin oxidoreductase
-_         Atu0028    17933953      31670..31831      53aa  hypothetical protein
+_         Atu0042    17933967      43469..43774     101aa  hypothetical protein
-_         Atu0056    17933978      59238..59627     129aa  hypothetical protein
+_         Atu0058    17933979      62235..62492      85aa  hypothetical protein
+_         Atu0067    17933986      70385..70729     114aa  hypothetical protein
+nrdH      Atu0068    17933987      70923..71174      83aa  glutaredoxin rotein
+nrdI      Atu0069    17933988      71188..71586     132aa  hypothetical rotein
+nrdF      Atu0071    17933990      73777..74751     324aa  ribonucleotide diphosphate reductase beta subunit
-_         Atu0076    17933995      78443..79132     229aa  hypothetical protein
+_         Atu0083    17934002      86610..86831      73aa  hypothetical protein
+_         Atu0094    17934013      98397..99086     229aa  hypothetical protein
-_         Atu0105    17934024     109363..109557     64aa  hypothetical protein
+_         Atu0107    17934026     110062..110223     53aa  hypothetical protein
+_         Atu0115    17934034     120729..120932     67aa  hypothetical protein
+_         Atu0116    17934035     121502..121912    136aa  hypothetical protein
+_         Atu0117    17934036     122012..122473    153aa  hypothetical protein
+_         Atu0118    17934037     122483..123988    501aa  hypothetical protein
+_         Atu0144    17934062     148441..148662     73aa  hypothetical protein
```

Figura A.5: Trecho do arquivo `atml.exc`. As colunas informam da esquerda para a direita, a identificação do gene; o sinônimo do gene; o identificador no GenBank; as coordenadas de DNA; o tamanho do gene e o nome do produto.

A Figura A.6 mostra um trecho do arquivo de saída de extensão **k12** resultante da comparação entre os genomas das Alphaproteobacterias *Agrobacterium tumefaciens str. C58* (at) e *Sinorhizobium meliloti 1021* (sm).

Gene	Synonym	start	product	best hit e-value	[regions]
-	Atu0001	203	hypothetical protein	1e-118	[1]
maf	Atu0002	1056	Maf like protein	1e-66	[1]
aroE	Atu0003	1648	shikimate 5 dehydrogenase	1e-112	[1]
coaE	Atu0004	2505	dephospho CoA kinase	2e-58	[1]
dnaQ	Atu0005	3110	DNA polymerase III subunit epsilon	3e-88	[1]
secB	Atu0006	3907	export protein SecB	2e-69	[1]
fxsA	Atu0007	4493	hypothetical protein	2e-28	[1]
-	Atu0008	5215	hypothetical protein	1e-88	[1]
mltA	Atu0009	5865	membrane bound lytic murein transglycosy	1e-131	[1]
-	Atu0010	6974	hypothetical protein	2e-53	[1]
-	Atu0011	7540	transcriptional regulator	2e-45	[1]
gyrB	Atu0012	8049	DNA gyrase subunit B	0	[1]
-	Atu0013	10604	NAD(P)H flavin oxidoreductase	6e-61	[1]
hpcE	Atu0014	11335	2 hydroxyhepta 2,4 diene 1,7 dioate isom	1e-26	[isolated]
depA	Atu0015	12279	intracellular PHB depolymerase	0	[2]
-	Atu0016	13605	hypothetical protein	5e-90	[2]
trpF	Atu0017	14475	N (5 phosphoribosyl)anthranilate isomer	3e-85	[2]
trpB	Atu0018	15139	tryptophan synthase subunit beta	0	[2]
trpA	Atu0019	16376	tryptophan synthase subunit alpha	1e-125	[2]
accD	Atu0020	17231	acetyl CoA carboxylase beta subunit	1e-144	[2]
folC	Atu0021	18214	folylpolyglutamate synthase	0	[2]
trxA	Atu0022	19641	thioredoxin C 1	6e-52	[2]
uvrD	Atu0023	20039	ATP dependant DNA helicase	0	[2]
-	Atu0024	23586	hypothetical protein	0	[2]
-	Atu0025	26768	hypothetical protein	1e-95	[2]
-	Atu0026	27520	hypothetical protein	1e-164	[2]
-	Atu0027	29028	two component sensor kinase	0	[2]
-	Atu0028	31670	hypothetical protein	===> NO HITS	
ahcY	Atu0029	31882	S adenosyl L homocysteine hydrolase	0	[2]
ptsH	Atu0030	33411	phosphocarrier protein HPr	2e-33	[2]
-	Atu0031	33698	PTS system, IIA component	3e-63	[2]
-	Atu0032	34225	hypothetical protein	1e-28	[2]

Figura A.6: Trecho do arquivo `atsm.k12` resultante da comparação entre *Agrobacterium tumefaciens str. C58* e *Sinorhizobium meliloti 1021*. As colunas informam da esquerda para a direita o nome do gene; seu sinônimo; a coordenada de início; o nome do produto; o e-value do melhor hit, se o gene possuir hits; e as regiões ortólogas das quais o gene faz parte.

A Figura A.7 mostra um trecho do arquivo de saída de extensão **mul** resultante da comparação entre os genomas *Xanthomonas axonopodis pv. citri str. 306* (xac) e *Xanthomonas campestris pv. campestris str. ATCC 33913* (xcc).

```
XAC 4312
XCC 4181
117
XACXCC20060502-1-Rc
1 36
1 36
42
36 36 1
35 34 1
34 33 1
33 32 1
32 31 1
31 29 1
30 28 1
30 27 0
30 26 0
29 28 0
29 27 1
29 26 0
28 28 0
28 27 0
28 26 1
27 25 1
26 24 1
25 23 1
24 22 1
```

Figura A.7: Trecho do arquivo `xacxcc.mul` resultante da comparação entre *Xanthomonas axonopodis pv. citri str. 306* e *Xanthomonas campestris pv. campestris str. ATCC 33913*. Nas duas primerias linhas do arquivo temos o prefixo do nome dos genomas em comparação e a quantidade de orfs em cada genoma. Na linha seguinte temos a quantidade de ROs no arquivo. Para cada RO temos o código da região; os genes de início e fim da região em cada genoma; o número de *matches* encontrados nessa região e os *matches* em cada linha. Cada linha contendo os *matches* informam da esquerda para a direita, o gene do primeiro genoma; o gene do segundo genoma e se o *match* é um BBH.

A Figura A.8 mostra um trecho do arquivo de saída de extensão **reg** resultante da comparação entre os genomas *Xanthomonas axonopodis pv. citri str. 306* e *Xanthomonas campestris pv. campestris str. ATCC 33913*.

```
>XACXCC2006036-55-Rc
4 matches
2kb in XAC - 3kb in XCC
=====================================================================
Gene    Synonym (XAC)    gi        size   product
=====================================================================
+_      XAC2122       21242857    274aa   dehydrogenase
+_      XAC2123       21242858    217aa   hypothetical protein
+_      XAC2124       21242859    200aa   hypothetical protein
+gtrB   XAC2125       21242860    237aa   glycosyl transferase related protein
=====================================================================
Gene    Synonym (XCC)    gi        size   product
=====================================================================
-gtrB   XCC1643       21231096    239aa   glycosyl transferase related protein
-_      XCC1644       21231097    198aa   hypothetical protein
-_      XCC1645       21231098    226aa   hypothetical protein
-_      XCC1646       21231099    332aa   dehydrogenase

=======
matches
=======
=============================================================================================
 Gene   Synonym     start size e-value [       best hit        ] product
=============================================================================================
+gtrB   XAC2125   2481258 237    1e-58 [best                    ] glycosyl transferase related protein
-gtrB   XCC1643   1918110 239    1e-58 [best                    ] glycosyl transferase related protein
---------------------------------------------
+_      XAC2124   2480704 200    7e-55 [best                    ] hypothetical protein
-_      XCC1644   1918784 198    7e-55 [best                    ] hypothetical protein
---------------------------------------------
+_      XAC2123   2480054 217    4e-56 [best                    ] hypothetical protein
-_      XCC1645   1919377 226    5e-56 [best                    ] hypothetical protein
---------------------------------------------
+_      XAC2122   2479137 274    8e-64 [best                    ] dehydrogenase
-_      XCC1646   1920126 332    1e-63 [best                    ] dehydrogenase
---------------------------------------------
```

Figura A.8: RO resultante da comparação entre os genomas dos organismos *Xanthomonas axonopodis pv. citri str. 306* (Xac) e *Xanthomonas campestris pv. campestris str. ATCC 33913* (Xcc) Na primeira linha do arquivo temos o código do RO; na linha seguinte temos o número de *matches* na RO; em seguida temos o tamanho aproximado (em número de bases) da RO em cada genoma. Para cada genoma, é mostrada uma linha contendo os genes que participam da RO. Em seguida cada par participante da RO é exibido. As colunas informam da esquerda para a direita, a fita e o nome do gene; o sinônimo do gene; a coordenada de início; o tamanho; o e-value do melhor hit; a palavra "best" na linha de cima do par indica que o gene de cima encontrou o gene de baixo do par como melhor hit, acontecendo o mesmo para a palavra "best" na linha de baixo do par; Em seguida temos o nome do produto do gene.

A Figura A.9 mostra um trecho do arquivo de saída de extensão **rns** resultante da comparação entre os genomas das Alphaproteobacterias *Sinorhizobium meliloti 1021* (sm) e *Mesorhizobium loti MAFF303099* (ml).

```
>SMML20060527-222-Pc
# of matches: 4
2kb in SM - 2kb in ML
=============================================================================================
 Gene  |Synonym    start size  e-value [       best hit       ] product
=============================================================================================
-rpsI  |SMc01803  1350279 155    7e-64 [best                  ] 30S ribosomal protein S9
-_     |mll8456   6948034 160    4e-64 [best                  ] 30S ribosomal protein S9
-----------------------------------------
-rplM  |SMc01804  1350749 154    3e-74 [best                  ] 50S ribosomal protein L13
-_     |mll8458   6948518 154    1e-74 [best                  ] 50S ribosomal protein L13
-----------------------------------------
-_     |SMc01805  1351432 143    2e-35 [best                  ] hypothetical protein
-_     |mll8460   6949197 154    1e-35 [best                  ] hypothetical protein
-----------------------------------------
+_     |SMc01806  1352032 273    1e-97 [best                  ] enoyl CoA hydratase
+_     |mlr8461   6949816 273    6e-98 [best                  ] enoyl CoA hydratase
-----------------------------------------
```

Figura A.9: Exemplo de um run paralelo consistente entre os genomas dos organismos *Sinorhizobium meliloti 1021* (Sm) e *Mesorhizobium loti MAFF303099* (Ml). Na primeira linha do arquivo temos o código do run; na linha seguinte temos o número de *matches* no run; em seguida temos o tamanho aproximado (em número de bases) do run em cada genoma. Cada par mostrado é um *match* do run. As colunas informam da esquerda para a direita, a fita e o nome do gene; o sinônimo do gene; a coordenada de início; o tamanho; o e-value do melhor hit; a palavra "best" na linha de cima do par indica que o gene de cima encontrou o gene de baixo do par como melhor hit, acontecendo o mesmo para a palavra "best" na linha de baixo do par; Em seguida temos o nome do produto do gene.

A Figura A.10 mostra um trecho do arquivo de saída de extensão **sre** resultante da comparação entre os genomas das Alphaproteobacterias *Agrobacterium tumefaciens str. C58* (at) e *Sinorhizobium meliloti 1021* (sm).

```
>Region from 934 to 952
19 orfs
=====================================================================
  Gene   Synonym       start..end        product
=====================================================================
# +_      Atu0951     940649..941881     large terminase
# -_      Atu0952     942264..942923     hypothetical protein
# +_      Atu0953     943042..943350     hypothetical protein
# +gp34   Atu0954     943607..944773     phage head portal protein
# +_      Atu0955     944969..945289     hypothetical protein
# +gp35   Atu0956     945339..945911     phage prohead protease
# +gp36   Atu0957     945944..947212     phage phi C31 major capsid gp36 like protein
# -_      Atu0958     947184..948038     hypothetical protein
# +_      Atu0959     948215..948610     hypothetical protein
  -_      Atu0960     948618..949724     MFS permease
# -_      Atu0961     949983..950693     hypothetical protein
# -_      Atu0962     950881..951285     hypothetical protein
# -_      Atu0963     951278..952276     hypothetical protein
# +_      Atu0964     952262..952900     hypothetical protein
# +_      Atu0965     952897..953781     hypothetical protein
# +_      Atu0966     953778..954212     hypothetical protein
# +_      Atu0967     954230..958030     hypothetical protein
# +_      Atu0968     958065..958271     hypothetical protein
# +_      Atu0969     958346..958645     hypothetical protein
```

Figura A.10: Região específica do genoma do organismo *Agrobacterium tumefaciens str. C58* (At) em relação ao genoma do organismo *Sinorhizobium meliloti 1021* (Sm). Inicialmente temos as coordenadas de início e fim da RE e na linha seguinte, temos a quantidade de genes pertencentes a RE. Cada proteína predita marcada com `#` é uma proteína predita específica.

A Figura A.11 mostra o arquivo `cbbh.pdf` que ilustra graficamente o alinhamento obtido pelo LCS entre os BBHs dos organismos *Xanthomonas axonopodis pv. citri str. 306* e *Xanthomonas campestris pv. campestris str. ATCC 33913*.

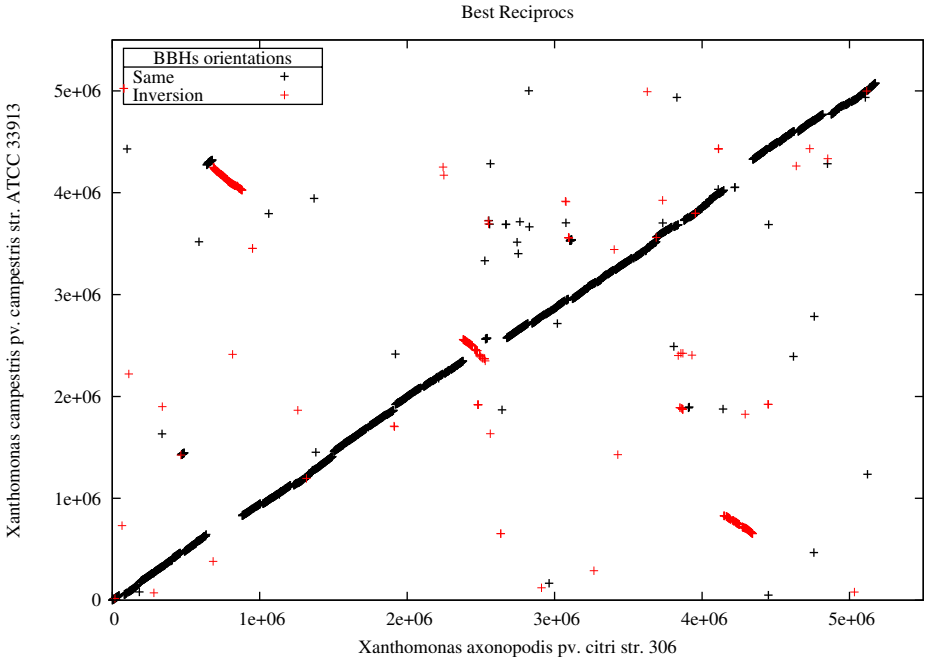


Figura A.11: Plotagem do alinhamento obtido pelo LCS entre os BBHs dos genomas dos organismos *Xanthomonas axonopodis pv. citri str. 306* (Xac) e *Xanthomonas campestris pv. campestris str. ATCC 33913.* (Xcc)

A Figura A.12 mostra o arquivo `hbbh.pdf` que ilustra graficamente os BBHs entre os organismos *Xanthomonas axonopodis pv. citri str. 306* e *Xanthomonas campestris pv. campestris str. ATCC 33913*.

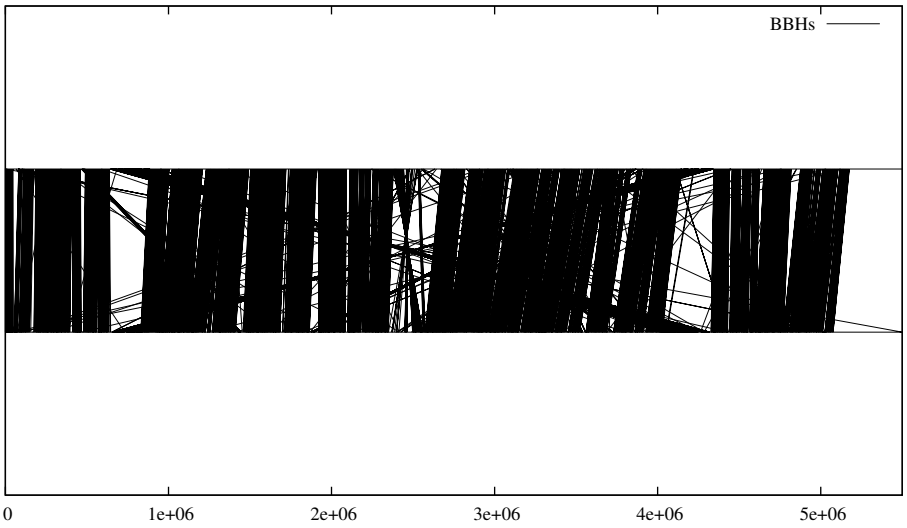


Figura A.12: Plotagem dos BBHs entre os genomas dos organismos *Xanthomonas axonopodis pv. citri str. 306* (Xac) e *Xanthomonas campestris pv. campestris str. ATCC 33913* (Xcc).

Apêndice B

Detalhes operacionais para obter parálogos

A nova proposta de metodologia apresentada na Seção 4.1 para tentar determinar as famílias de genes parálogos, resultou na implementação de um conjunto de programas e scripts pertencentes ao pacote denominado PARALOGS. Descreveremos nas seções seguintes alguns detalhes dos programas e scripts desse pacote.

B.1 Pacote PARALOGS

O pacote PARALOGS contém um conjunto de programas e scripts que foram escritos na linguagem `C++` e `Perl`. Os programas escritos em `C++` utilizaram o compilador `g++` (GNU software).

Conforme descrito na Seção 4.1, a metodologia consiste de duas fases (Agrupamento de seqüências e Busca homólogos). Os programas e scripts do pacote PARALOGS iniciam no segundo passo da fase "Agrupamento de seqüências", que consiste em construir um grafo G e determinar as cliques maximais nesse grafo. O primeiro passo dessa fase é determinado pelo programa EGG.

Os programas do pacote PARALOGS pertencentes a fase "Agrupamento de seqüências" e suas funcionalidades são descritas a seguir:

- `cria_arestas` - programa que cria arestas de um grafo, a partir do arquivo `gGgG.12`, resultante da comparação do genoma `gG` com ele mesmo. Os vértices do grafo são os genes do genoma `gG`, e uma aresta entre os genes v e w existe, se v encontrou w como hit e vice-versa, conforme descrito em 4.1.1. O programa contém ainda os parâmetros *e-value* e *cobertura*, que são utilizados para determinar se uma aresta existe entre dois vértices de `gG`. A saída é um arquivo texto de extensão `grafo_original` contendo o grafo no formato de lista de adjacências, como entrada para o programa que encontrará as componentes conexas nesse grafo.

- `componentes` - programa gerador de componentes conexas, a partir do arquivo de extensão `grafo_original` (saída de `cria_arestas`). O programa cria o arquivo

de extensão `comps` contendo o grafo original sem os vértices de grau zero. Cada componente é criada em um arquivo texto de extensão `comp` com o nome composto pelo nome do arquivo de extensão `grafo_original` e com um dígito indicador do número da componente. A criação do arquivo `comps` objetiva a utilização dessa saída em um outro programa que "lê" componentes separadamente.

- `cliques` - programa que encontra as cliques maximais do subgrafo do grafo original que representa o genoma, a partir do arquivo `gGg.comps` (saída de `componentes`). A saida consiste em arquivos textos com extensão `cfc`. Cada arquivo das cliques são identificados pelo dígito indicador do número da clique.

Exemplos de alguns arquivos de saída de cada programa listado na relação acima podem ser vistos em B.2.

A implementação da fase "Busca Homólogos" 4.1.2 foi realizada utilizando os seguintes programas e scripts:

- `cria_multi_fasta.pl` - script que cria arquivos multi-fasta, sem as seqüências discrepantes, a partir dos arquivos das cliques maximais (saída do programa `cliques`). Para cada arquivo de entrada que passou pelo teste das seqüências discrepantes, o script gera um arquivo de saída com extensão `faa` contendo as seqüências no formato FASTA. Para cada arquivo de extensão `faa`, um arquivo de extensão `fqf` será criado, e armazenará inicialmente as proteínas nativas e posteriormente, as proteínas agregadas. Um parâmetro importante desse script é o número mínimo de proteínas pertencentes a uma determinada família. O valor padrão é 2.

- `cria_hmms_de_cliques.pl` - script que cria os modelos ocultos de Markov para cada arquivo multi-fasta gerado por `cria_multi_fasta.pl`. A saída desse escript são os arquivos de extensão `aln`, `dnd` e `hmm`, resultantes da utilização dos programas `clustalW` e `hmmbuild`. Em seguida utilizamos o programa `hmmcalibrate` para cada arquivo de extensão `hmm` para calibrar o determinado modelo.

- `busca_faa_hmm.pl` - script que agrega proteínas às novas famílias a partir dos arquivos de extensão `fqf` e `hmm`. O script gera para cada família que passou pelo teste de discrepantes (arquivos de extensão `fqf`), um arquivo texto de extensão `ff` contendo informações das proteínas nativas e agregadas; e mais a saída do arquivo `clustalW` gerada para a família.

Na Figura 4.2 temos uma ilustração do fluxo de execução da fase "Busca Homólogos". Trechos dos arquivos de saída de cada script listado na relação acima podem ser vistos na seção seguinte.

B.2 Exemplos de alguns arquivos de saída

Para ilustrarmos os exemplos dos arquivos de saída da seção anterior, utilizamos os genomas *Pseudomonas aeruginosa* PA01 (Pa) e *Synechocystis sp.* PCC 6803 (Sp) que foram utilizados em [5].

As figuras B.1 e B.2 mostram, respectivamente, o conteúdo do arquivo `papa.10.comp` e um trecho do arquivo `papa.comps`, resultantes da execução do programa `componente` para o genoma *Pseudomonas aeruginosa*.

```
6
30
3234
5091
5098
5373
5383
```

Figura B.1: Componente conexa encontrada no grafo gerado a partir das proteínas do genoma do organismo *Pseudomonas aeruginosa* (Pa). A primeira linha indica o número de proteínas pertencentes a essa componente. Cada linha seguinte indica o índice da proteína no arquivo de extensão `ptt`.

```
5565
1    -1   0
2    -1   0
3    -1   0
4     1   1   4962
5    -1   0
6    -1   0
7     2   3   2076 1939 2075
8    -1   0
9    -1   0
10   -1   0
11    3   1   3240
12   -1   0
13   -1   0
14   -1   0
15   -1   0
16   -1   0
17   -1   0
18    4   1   943
19    5   1   1121
20   -1   0
21   -1   0
22    6   1   3197
23    7  12   2195 3627 862 1647 2117 5422 3254 2489 3565 1136 5229 2678
24   -1   0
25    8   1   243
26   -1   0
27   -1   0
28   -1   0
29    9   3   2561 103  1646
30   10   3   3234 5383 5373
```

Figura B.2: Trecho do arquivo `papa.comps`. A primeira linha do arquivo indica o número de proteínas encontradas no arquivo de extensão `ptt`. As linhas seguintes indicam da esquerda para a direita, o índice da proteína no arquivo de extensão `ptt`; o número da componente conexa que a protéina pertence (-1 se não pertencer a nenhuma); a quantidade de proteínas vizinhas na componente; e os índices, segundo o arquivo `ptt`, das proteínas vizinhas na componente.

As figuras B.3 e B.4 mostram o conteúdo dos arquivos `spsp.155.cfc` e `spsp.155.fqf` resultantes, respectivamente, da execução do programa `cliques` e `multi_fasta` para o genoma *Synechocystis sp.*

```
4
262
2592
2829
2173
```

Figura B.3: Clique maximal encontrada no grafo gerado para o genoma do organismo *Synechocystis sp.* (Ssp).

```
3
262
2592
2829
```

Figura B.4: Família encontrada após eliminar as seqüências discrepantes de sua clique maximal.

A Figura B.5 mostra um trecho do arquivo `spsp.155.ff` resultante da execução do script `busca_faa_hmm.pl` para o genoma *Synechocystis sp.*.

```
  1651914     1524    1233    hypothetical protein
  1001724     1629    1233    phytoene dehydrogenase
  1001311     1506    1233    hypothetical protein
# 1001515     1416    1231    hypothetical protein


CLUSTAL W (1.83) multiple sequence alignment


gi|1651914|dbj|BAA16840.1|      --------------MVPSNAESQSVVVIGAGIGGLTTAALLAQQGYRVKV
gi|1001311|dbj|BAA10798.1|      ----------------MTVSPSYDAIVIGSGIGGLVTATQLVSKGLKVLV
gi|1001724|dbj|BAA10561.1|      --------------------MITTDVVIIGAGHNGLVCAAYLLQRGLGVTL
gi|1001515|dbj|BAA10142.1|      MVIRSGKTNLNPPCALMAPSSSCDCIIVGSGLSGLIAARNLSRVNYSVLV
                                                    : . :::*:* .**  *  *   .  * :

gi|1651914|dbj|BAA16840.1|      YEQAAIVGGCASTFRRRGFIFDVGATQVAGLEPGGIHH-RIFRQLQVD-L
gi|1001311|dbj|BAA10798.1|      LERYLIPGGSAGYFEREGYRFDVGASMIFGFGDRGTTN-LLTRALAAVGQ
gi|1001724|dbj|BAA10561.1|      LEKREVPGGAATTEALMPELSPQFRFNRCAIDHEFIFLGPVLQELNLAQY
gi|1001515|dbj|BAA10142.1|      IEAQERLGG-----RMYGEYLPSGQWIDRGGQWVGPTQDRFLALLNEYNI
                                 *     **                   .          .   *

gi|1651914|dbj|BAA16840.1|      -IVGGVG-QRLTTFGPFGFAP--RTPIRNLWLVGDSVHPGEG--TAGVSY
gi|1001311|dbj|BAA10798.1|      -TYGPIPRRRLPGLLPMPFN---RTAIPGLYCVGDSTFPGQG--LNAVAF
gi|1001724|dbj|BAA10561.1|      VYHLDMSLDQMMFLRPLPEIANYQTPIKNLYLTGAGTHPGGS--ISGMP-
gi|1001515|dbj|BAA10142.1|      -WVGGGYAAFMPPGVWTSFGQALSAPVGRIHWAGTEIAPRWAGFFDGAIR
                                          :            :.:  :  .*    *  .    .

gi|1651914|dbj|BAA16840.1|      PQGLWHLQGSMQTLGDRLVEALKNHGGELFCSQRVEQIHCGQGYVEGVTV
gi|1001311|dbj|BAA10798.1|      G-GINYPKGGVGQIAESLVAGLEKFGGKIRYGARVTKIIQENNQAIGVEL
gi|1001724|dbj|BAA10561.1|      LEGIARPKGGTGALTEALVKLVQAQGGKILTDQTVKRVLVENNQAIGVEV
gi|1001515|dbj|BAA10142.1|      NPEAELLHGGAGQIPQKIAAELGN---SILLGEPVIHIAQDDK---GVEV
                                       :*.   : : :.  :     .:  .  * ::   :    ** :
```

Figura B.5: Família encontrada após agregar uma nova proteína. A parte inicial do arquivo consiste das informações sobre a família. Essas informações indicam da esquerda para a direita; o identificador da proteína no GenBank; o tamanho da proteína; o número COG da classificação da proteína; e o nome do produto. Abaixo dessas informações, temos a concatenação do arquivo de saída resultante da execução do programa `clustalW` para as seqüências da família.

Referências Bibliográficas

[1] N. F. Almeida. *Ferramentas para comparação genômica*. Tese de doutorado, IC-UNICAMP, 2002.

[2] N. F. Almeida e J. C. Setubal. Um modelo oculto de markov para encontrar promotores em seqüências de DNA. Relatório técnico, Institute of Computing, University of Campinas, Brazil, 1998.

[3] N. F. Almeida e J. C. Setubal. A set of tools for detailed syntatic pairwise comparison of whole bacterial genomes, 2000. Manuscript.

[4] N. F. Almeida e J. C. Setubal. Detection of related genes in procaryotes using syntenic regions. In *DIMACS Workshop on Whole Genome Comparison*. DIMACS Center, Rutgers University, February 2001.

[5] N. F. Almeida, J. C. Setubal, M. H. Carvalho, e C. J. M. Viana. A method to determine homologous protein families based on graph cliques and profiles. In *3th Brazilian Workshop on Bioinformatics*. October 2004. WOB Brasília (poster).

[6] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, e D. J. Lipman. A basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[7] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, e D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acid Research*, 25:3389–3402, 1997.

[8] C. E. R. Alves, E. N. Cáceres, e S. W. Song. A BSP/CGM algorithm for finding all maximal contiguous subsequence of a sequence of numbers. EUROPAR, 2006.

[9] R. V. Andrade, M. E. M. T. Walter, M. J. A. Carvalho, N. F. Almeida, M. M. Brígido, e M. S. S. Felipe et al. Overview and perspectives on the transcriptome of paracoccidioides brasiliensis. *Revista Iberoamericada de Micología*, 22:203–212, 2005.

[10] G. S. Araújo. *Filogenia de Proteomas*. Dissertação de Mestrado, DCT-UFMS, 2003.

[11] A. Bateman, E. Birney, L. Cerruti, R. Durbin, L. Etwiller, S. R. Eddy, S. Griffiths-Jones, K. L. Howe, M. Marshall, e E. L. Sonnhammer. The pfam protein families database. *Nucleic Acids Research*, 30:276–280, 2002.

[12] M. M. Brígido, M. M. T. Walter, A. G. Oliveira, M. K. Inoue, D. S. Anjos, E. F. O. Sandes, J. J. Gondim, M. J. A. Carvalho, N. F. Almeida, e M. S. S. Felipe. Bioinformatics of the P*aracoccidioides brasiliensis* EST Project. *Genetics and Molecular Research*, 4:203–215, 2005.

[13] S. B. Cannon e N. D. Young. OrthoParaMap: Distinguishing orthologs from paralogs by integrating comparative genome data and gene phylogenies. *BMC Bioinformatics*, 4, 2003.

[14] T. H. Cormen, C. E. Leiserson, e R. L. Rivest. *Introduction to Algorithms*. MIT Press/McGraw-Hill, Cambridge,MA/New York, 1990.

[15] A. C. Rasera da Silva, J. C. Setubal, e N. F. Almeida et al. Comparison of the genomes of two *xanthomonas* pathogens with differing host specificities. *Nature*, 417(6887):459–463, May 2002.

[16] A. L. Delcher, S. Kasif, R. D. Fleischmann, J. Peterson, O. White, e S. L. Salzberg. Alignment of whole genomes. *Nucleic Acids Research*, 27:2369–2376, 1999.

[17] S. R. Eddy. Profile hidden markov models. *Bioinformatics*, 14:755–763, 1998.

[18] S. R. Eddy. What is a hidden markov model? *Nature Biotechnology*, 22:1315–1316, 2004.

[19] A. J. Enright, S. V. Dongen, e C. A. Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Research*, 30:1575–1584, 2002.

[20] M. S. S. Felipe, M. E. M. T. Walter, M. M. Brígido, e N. F. Almeida et al. Transcriptome characterization of the dimorphic and pathogenic fungus paracoccidioides brasiliensis by est analysis. *Yeast*, 20(3):263–271, 2003.

[21] M. S. S. Felipe, M. E. M. T. Walter, M. M. Brígido, e N. F. Almeida et al. Transcriptional profiles of the human pathogenic fungus paracoccidioides brasiliensis in mycelium and yeast cells. *Journal of Biological Chemistry*, 280(26):24706–24714, 2005.

[22] M. Kellis. *Computational Comparative Genomics: Genes, Regulation, Evolution*. Tese de doutorado, Massachusetts Institute of Technology, 2003.

[23] M. Kellis, N. Patterson, B. Birren, B. Berger, e E. S. Lander. Methods in comparative genomics: genome correspondence, gene identification and motif discovery. *Journal of Computational Biology*, 11:319–355, 2004.

[24] C. B. Monteiro-Vitorello, L. E. A. Camargo, J. C. Setubal, e N. F. Almeida et al. The genome sequence of the gram-positive sugarcane pathogen leifsonia xyli subsp. xyli. *Molelular Plant-Microbe Interactions*, 17(8):827–836, 2004.

[25] L. Montera. *Regiões Ortólogas em Múltiplos Genomas*. Dissertação de Mestrado, DCT-UFMS, 2004.

[26] L. M. Moreira, R. F. de Souza, N. F. Almeida, J. C. Setubal, J. C. F. Oliveira, L. R. Furlan, J. A. Ferro, e A. C. R da Silva. Comparative genomics analyses of citrus-associated bacteria. *Annual Reviews Phytopathology*, 42:163–184, 2004.

[27] Nomenclature Committee of the International Union of Biochemistry (NC-IUB). Nomenclature for incompletely specified bases in nucleic acid sequences.recomendations 1984. *Europen Journal of Biochemistry.*, 150:1–5, 1985.

[28] P. A. Pevzner. *Computational Molecular Biology*. MIT Press, 2000.

[29] S. Rudd. Expressed Sequence Tags: alternative or complement to whole genome sequences? *TRENDS in Plant Science*, 8:321–329, 2003.

[30] W. L. Ruzzo e M. Tompa. A linear time algorithm for finding all maximal scoring subsequences. In *Seventh International Conference on Intelligent Systems for Molecular Biology*. Heidelberg, Germany, 1999.

[31] S. L. Salzberg, A. L. Delcher, A. Phillippy, e J. Carlton. Fast algorithms for large-scale genome alignment and comparison. *Nucleic Acids Research*, 30:2478–2483, 2002.

[32] S. L. Salzberg, S. Kurtz, A. Phillippy, A. L. Delcher, M. Smoot, M. Shumway, e C. Antonescu. Versatile and open software for comparing large genomes. *Genome Biology*, 5, 2004.

[33] S. L. Salzberg, D. B. Searls, e S. Kasif, editores. *Computational Methods in Molecular Biology*, volume 32 de *New comprehensive biochemistry*. Elsevier, Netherlands, 1998.

[34] D. Sankoff. Gene and genome duplication. *Current Opinion in Genetics & Development*, 11:681–684, 2001.

[35] J. C. Setubal. Computational analyses fo bacterial genomes: tasks, techinques, and tools. Relatório técnico, Institute of Computing, University of Campinas, Brazil, 2002.

[36] J. C. Setubal e J. Meidanis. *Introduction to Computational Molecular Biology*. PWS Publishing Co., 1997.

[37] M. A. Van Sluys, J. C. Setubal, e N. F. Almeida et al. Comparative analyses of the complete genome sequences of pierce's disease and citrus variegated chlorosis strains of xylella fastidiosa. *Journal of Bacteriology*, 185(3):1018–1026, 2003.

[38] G. D. Stormo e G. W. Hartzell. Identifying Protein-Binding Sites from Unaligned DNA Fragments. *PNAS*, 86:1183–1187, 1989.

[39] J. Tamames. Evolution of gene order conservation in prokaryotes. *Genome Biology*, 2(6), 2001.

[40] R. L. Tatusov, M. Y. Galperin, D. A. Natale, e E. V. Koonin. The COG database: a tool for genome-scale analysis of protein functions and evolution. *Nucleic Acids Research*, 28:33–36, 2000.

[41] G. P. Telles, M. M. Brígido, N. F. Almeida, C. J. M. Viana, D. A. S. Anjos, e M. E. M. T. Walter. A method for comparing three genomes. *Lecture Notes in Computer Science*, 3594:160–169, 2005.

[42] J. D. Thompson, D. G. Higgins, e T. J. Gibson. ClustalW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22:4673–4680, 1994.

[43] A. T. R.de Vasconcelos, C. A. M. Russo, e C. J. M. Viana et al. Mammibase: a mammalian mitochondrial genome database. In *2nd International Conference on Bioinformatics and Computational Biology*. October 2004. Icobicobi - Angra dos Reis, RJ (poster).

[44] A. T. R.de Vasconcelos, C. A. M. Russo, e C. J. M. Viana et al. Mammibase: a mitochondrial genome database for mammalian phylogenetic studies. *Bioinformatics*, 21:2566–2567, May 2005.

[45] D. W. Wood, J. C. Setubal, N. F. Almeida, e et al. Sequencing and analysis of the agrobacterium tumefaciens genome. In *10th Int'l congress on Molecular plant-microbe interactions*. 2001. Madison, WI (poster).

[46] D. W. Wood, J. C. Setubal, e N. F. Almeida et al. The genome of the natural genetic engineer *agrobacterium tumefaciens* c58. *Science*, 294(5550):2317–2323, December 2001.

[47] G. G. Zerlotini, D. A. S. Anjos, M. E. M. T. Walter, M. M. Brígido, G. P. Telles, C. J. M. Viana, e N. F. Almeida. A method for obtaining orthologous genes among three genomes. 2006. In Preparation.

[48] Documentação on-line de EGG. Disponível em URL: `http://egg.dct.ufms.br/egg`. Acesso em 14 de julho de 2006.

[49] National center for biotechnology information - NCBI. Disponível em URL: `http://www.ncbi.nlm.nih.gov/`. Acesso em 5 de junho de 2006.

[50] STRING - Search Tool for the Retrieval of Interacting Genes/Proteins. Disponível em URL: `http://string.embl.de/`. Acesso em 10 de agosto de 2006.