PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 02/11

Protein World Database: Definição e Implementação de Estruturas Organizacionais

Carlos Juliano Moura Viana

Sérgio Lifschitz

Antonio Basílio de Miranda

Edward Hermann Haeusler

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900

RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 02/11 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena Abril, 2011

Protein World Database: Definição e implementação de Estruturas Organizacionais

Carlos Juliano Moura Viana, Sérgio Lifschitz, Edward Hermann Haeusler e Antonio Basílio de Miranda

cviana@inf.puc-rio.br, sergio@inf.puc-rio.br, herman@inf.puc-rio.br, antonio@fiocruz.br

Abstract. The fast development of new genome sequencing technologies are contributing to increase the scale and resolution of many genomic comparative studies. In this way, the use of computational analysis techniques are becoming indispensable tools for a better understanding of relationships between organisms being studied. The main challenge faced by many researchers is the analysis of the data obtained from sequence alignments in order to obtain a better characterization of the studied organisms (characterizations in terms of their biological features, and also their relationships with the environment). This work aims to contribute to the context of genomic analysis field by modifying the methodology and implementation of some genomic sequencies comparison techniques, and in this way to help the research Genome Comparison Project - GCP[1]. The contribution lies in the modification done to include the way how the orthologous/specific genes and regions are constructed, and also in the way how database techniques are used to optimize access and use of data from sequence comparisons of more than 400 organisms.

Keywords: Genome Sequence Comparison, Protein World DB, Ortholog Region Gene Prediction

Resumo. O rápido desenvolvimento de novas tecnologias de sequenciamento estão estendendo substancialmente a escala e resolução de muitos trabalhos de genômica comparativa, tornando a utilização de técnicas computacionais de análise, ferramentas indispensáveis para uma melhor compreensão dos relacionamentos entre os organismos em estudo. O desafio central enfrentado por muitos pesquisadores consiste na análise dos dados de alinhamento de sequência, buscando uma melhor caracterização dos organismos estudados, tanto em termos de suas funcionalidades biológicas, quanto no relacionamento dessas funcionalidades com o ambiente onde se encontram. O presente trabalho está inserido no contexto de análise de genomas, especificamente consiste na alteração da metodologia e implementação de determinadas técnicas de comparação de sequências genômicas, com a finalidade de aplicação ao projeto de pesquisa de Comparação de Genomas - GCP [1]. A reformulação das metodologia inclui a alteração da construção das Estruturas Organizacionais de genes e regiões ortólogas/específicas entre dois genomas, assim como a utilização de técnicas de bancos de dados para otimizar o acesso e utilização dos dados de resultados de comparações de sequências de mais de 400 organismos.

Palavras-chave: Comparação de Sequências Genômicas, Protein World DB, Predição de Regiões e Genes Ortólogos

Responsável por publicações:
Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

Sumário

Lista de figuras

1 Introdução

O Projeto de Comparação de Genomas [1] ou GCP, sigla em inglês para *Genome Comparison Project*, é um projeto de pesquisa que resultou da união de esforços entre as equipes do Laboratório de Genômica Funcional e Bioinformática do Instituo Oswaldo Cruz (IOC) - Fiocruz; do laboratório de Bioinformática (LaBBio) - PUC-Rio e IBM. Esse projeto foi desenvolvido com o objetivo de comparar informações de proteínas em escala genômica, com a finalidade de melhorar a qualidade e interpretação dos dados biológicos e também nossa compreensão dos sistemas biológicos e suas interações. Essas comparações foram realizadas utilizando-se o programa SSEARCH [2; 3], uma implementação do rigoroso algoritmo local de programação dinâmica de Smith-Waterman [4].

As sequências utilizadas nas comparações foram obtidas da base de dados do RefSeq, versão 21, consistindo de todas as proteínas preditas que estavam codificadas em 485 genomas, completamente sequenciados e não completamente sequenciados, e da base de dados do Swiss-Prot, versão 51.5, onde foram obtidas 254.609 sequências de proteínas.

Para a comparação desse conjunto de proteínas, suas sequências foram organizadas em blocos, com no máximo 2000 sequências por bloco, e dessa forma, foram realizadas mais de um milhão de comparações na forma para-a-par, utilizando os valores padrões dos parâmetros do programa SSEARCH, com o valor de corte de *E-value* igual a 1.0.

Essa comparação resultou em diversos arquivos texto, um para cada comparação, contendo os identificadores das sequências, tamanho do alinhamento entre elas, coordenadas das regiões similares, a porcentagem de identidade, a quantidade de espaços preenchidos no alinhamento entre as sequências, os valores de pontuações referentes a similaridade entre as sequências e o valor estatístico de probabilidade *E-value*. A figura 1 ilustra um exemplo de um arquivo de relatório de comparação entre sequências dos arquivos multi-fastas **10000001.faa** e **10000142.faa**.

```
10000001.faa 10000142.faa
67904470,13474329,279,70.9,3.1e-013,0.231,481,247,708,9,455,19,34
67904470,13475325,194,51.1,2.7e-007,0.258,248,414,659,171,386,2,32
67904470,13475327,192,50.6,4e-007,0.297,212,446,654,202,397,3,16
67904470,13473997,115,32.8,0.059,0.238,239,45,263,45,280,20,3
```

Figura 1: Exemplo de relatório produzido pela comparação entre as sequências. Primeira linha indica os arquivos de origem das sequências comparadas.

A esse conjunto de dados, foram incluídos outros referentes a dados de anotações, tais como: características de genes e proteínas (banco do RefSeq), informações sobre a taxonomia (banco de taxonomia do NCBI [5]), dados sobre ontologias dos genes (banco Gene Ontology [6]), dados sobre domínio e famílias de proteínas (banco Pfam [7]) e de atividade enzimática (KEGG [8]).

Esse conjunto de dados foram modelados e armazenados em um banco de dados relacional, originando a primeira versão do banco de dados do *Protein World DB - PWD*, onde foram armazenados somente os dados das comparações que possuiam valor de *E-value* $\leq 10^{-3}$. Para armazenar e gerenciar esses dados foi utilizado o Sistema de Gerência de Banco de Dados (SGBD) IBM® DB2. Maiores detalhes sobre esse trabalho podem ser vistos em [9].

Posteriormente a essa versão, foram desenvolvidos outros trabalhos relacionados a re-estruturação do modelo conceitual e lógico para possibilitar alta disponibilidade, evitar perdas de desempenho, possibilitar também a execução eficiente de consultas, manutenção do banco de dados, e além de corrigir eventuais erros de modelagem. Dessa forma, Tristão e colegas [10; 11] propuseram um novo esquema conceitual e lógico para armazenamento e persistência desses dados biológicos, gerando assim uma nova versão (**PWD-V2**) para o banco de dados do PWD. As figuras 2 e 3 ilustram, respectivamente, o esquema conceitual e o esquema lógico, resultante do mapeamento realizado em [11], que utilizamos em nosso trabalho.

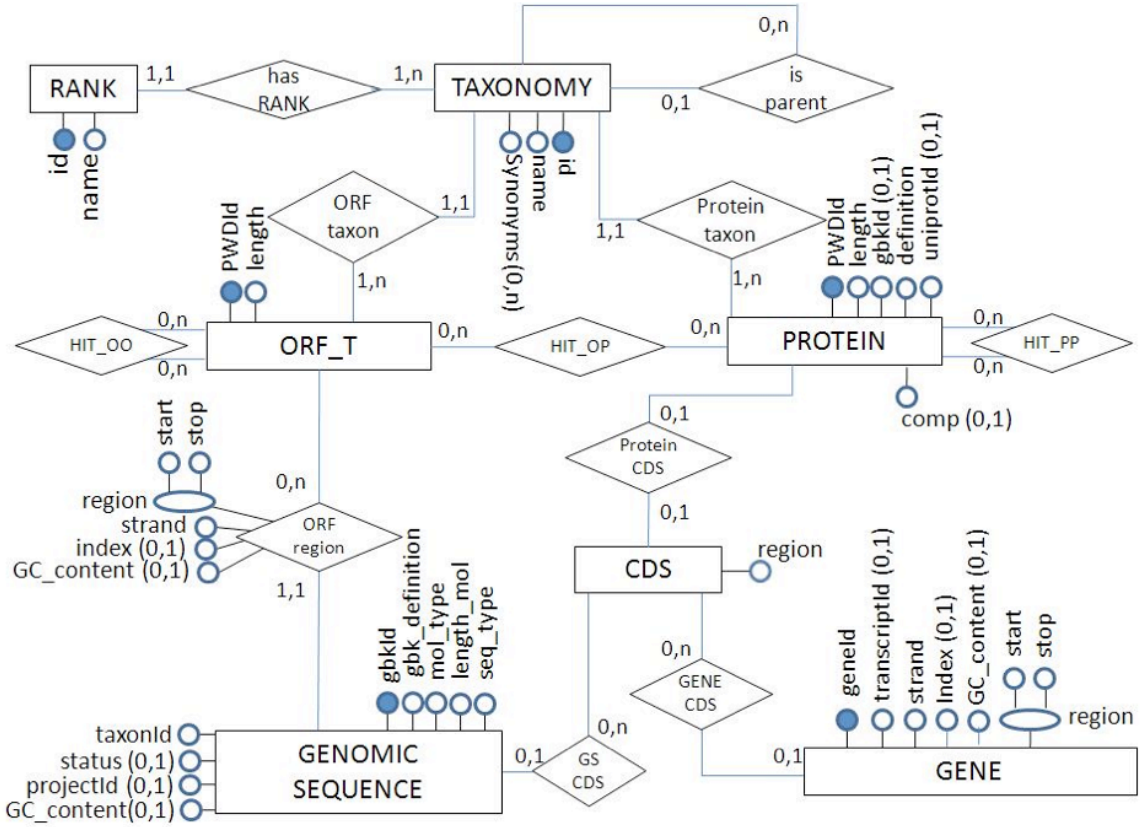

Figura 2: Esquema conceitual para a segunda versão do banco de dados do PWD.

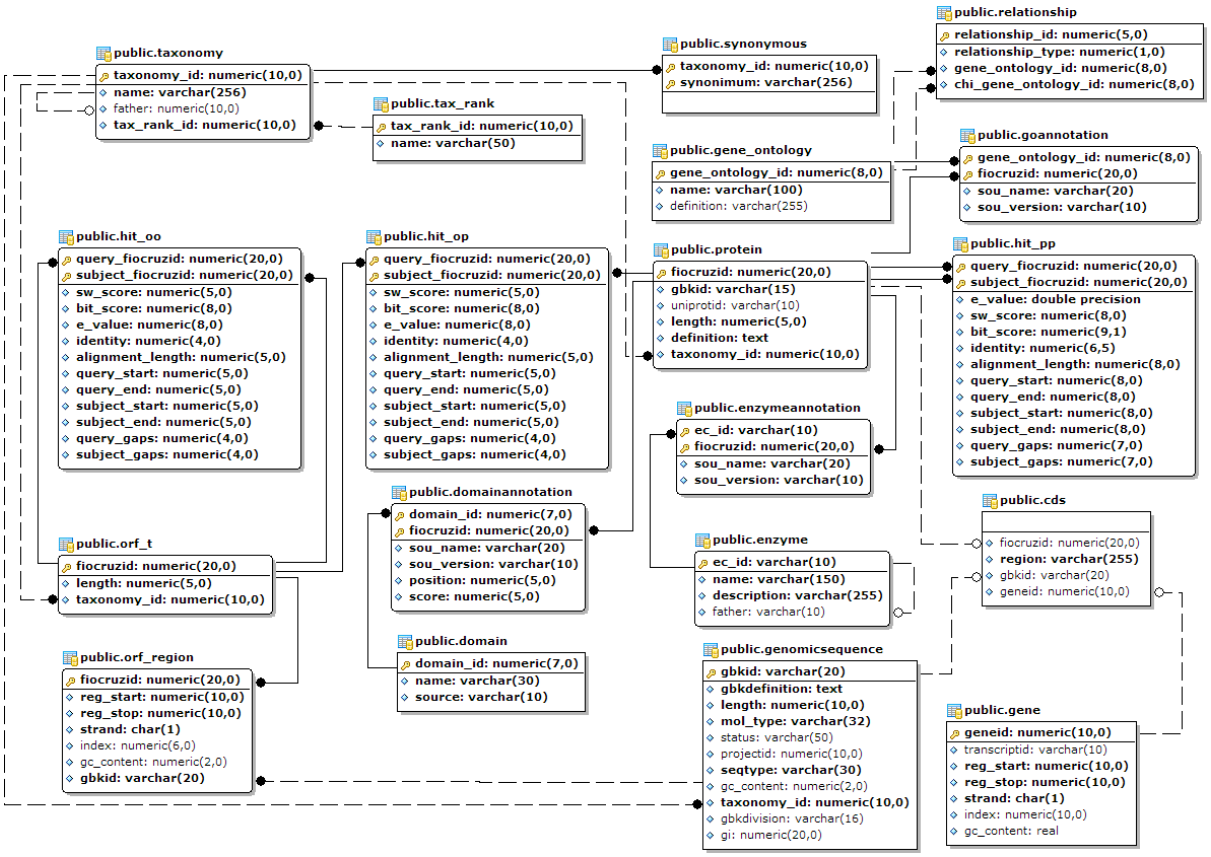

Figura 3: Esquema lógico para o banco de dados PWD-V2.

Para armazenar e gerenciar os dados, a partir dos novos esquemas, utilizou-se o SGBD PostgreSQL [12], versão 8.4. A escolha por esse SGBD foi devido a ser muito difundido entre a comunidade científica, além de ser um software livre e de código aberto. Maiores detalhes sobre a forma de importação e armazenamento dos dados dos relatórios das comparações entre as sequências podem vistos em [11].

Atualmente, as seguintes análises estão implementadas no PWD mantido pelo DB2:

1. Consulta das características de anotações por identifcadores das sequências, por termos do *GO*, por número EC da base do *KEGG* ou por termos do *Pfam*;

2. Consulta por todas as proteínas armazenadas que são similares a sequência de entrada. É permitido ao usuário filtrar os resultados pelos valores de *E-value*, porcentagem de sobreposição do alinhamento da sequência de entrada contra as sequências do banco de dados, entre outros filtros;

3. Comparação das proteínas não incluídas na base de dados contra todas as proteínas da base de dados, utilizando-se o programa BLAST [13; 14], entre outras consultas [9].

Embora as consultas descritas acima possam ser diretamente mapeadas para o novo esquema lógico, gerenciado pelo PostgreSQL, estas não são o objetivo geral desse trabalho, o qual está baseado em gerar outras consultas mais elaboradas (estruturas organizacionais), a fim de que auxiliem ainda mais na geração de conhecimento sobre as sequências armazenadas nessa base de dados.

Dessa forma, o presente trabalho está inserido no contexto de análise dos genomas presentes no banco de dados gerenciado pelo PostgreSQL, consistindo em tentar responder a algumas questões referentes aos resultados das comparações das sequências de proteínas dos genomas estudados. Estamos interessados em tentar responder questões relativas a:

1. Quais são as proteínas compartilhadas por duas ou mais espécies (inferir proteínas ortólogas);
2. Quais são as regiões de cada genoma que possuem essas proteínas compartilhadas;
3. Quais são as proteínas similares presentes em um mesmo organismo (inferir proteínas parálogas);
4. Determinar genes que são exclusivos a uma espécie em particular, ou seja, genes que estão restritos taxonomicamente;
5. Determinar as regiões de maior ocorrência desses genes exclusivos;
6. Determinar os genes exclusivos a um genoma, genes sem similaridade com outros genes, onde a distância taxonômica entre os genomas é maior ou igual a um determinado limiar;
7. Determinar grupos de proteínas relacionadas a partir de uma proteína de interesse ou por uma função biológica relevante.

Para responder a essas questões teremos que responder também outras perguntas mais simples, relacionadas a quantificação, anotação e seleção de dados específicos do banco de dados. Entre essas perguntas, podemos citar:

1. Questões estatísticas:
    - Quantos genomas e genes existem no banco de dados?
    - Quantos e quais são os genes pertencentes a uma determinada fita de cada genoma?
    - Quantos genomas pertencem a uma determinada classificação taxonômica específica?
2. Questões sobre seleção de determinados dados:
    - Quais são todas as proteínas para um determinado *Enzyme Commision number - EC* ec_id?
    - Quais são todos os hits para um determinado gene g_i com $E{-}value \leq threshold$?

Em específico, o trabalho consistem em redefinfir algumas consultas existentes e implementar novas outras consultas, funções e ou procedimentos abordando a perspectiva da criação de estruturas organizacionais, para que possam auxiliar na comparação dos genomas no nível dos seus genes e proteínas.

Na Seção 1 apresentamos uma introdução sobre o projeto de comparação de genomas, descrevendo o esquema conceitual e lógico utilizados pelo banco de dados do ProteinWorldDB, além de descrever as perguntas que queremos responder neste trabalho. Na Seção 2 apresentamos algumas definições e anotações que são utilizadas no decorrer deste trabalho. Alguns trabalhos relacionados são apresentados na Seção 3. Na Seção 4 descrevemos as modificações na metodologia e implementação das estruturas organizacionais. Descrevemos na Seção 5 algumas consultas mais simples que podemos fazer na base de dados. Na Seção 6 descrevemos alguns trabalhos futuros. Por fim, apresentamos a conclusão do nosso trabalho na Seção 7.

2 Definições Gerais

Nesta seção abordamos descrições de conceitos e notações utilizados nesse trabalho. Descrevemos alguns conceitos básicos de Biologia Molecular, tais como genomas, genes, proteínas e outros, além de alguns conceitos de Computação, tais como grafos e árvores. Esses conceitos são abordados de uma maneira simplificada e suas descrições são importantes para permitir uma melhor interpretação e compreensão das metodologias descritas nas seções posteriores.

2.1 Fundamentos básicos de Biologia Molecular

Neste trabalho faremos uso de alguns conceitos básicos de Biologia Molecular que estão presentes na literatura [15; 16; 17], entre outros. Em específico, utilizaremos os conceitos apresentados em [18; 19] para os termos de genomas, proteomas, genes, proteínas, homologia e domínios. Apesar de presumirmos no presente trabalho que o leitor tenha um conhecimento básico de Biologia Molecular, muitos dos conceitos desse domínio foram descritos a seguir para auxiliar na compreensão deste trabalho.

Neste trabalho utilizamos o termo **DNA** para referenciar uma sequência de letras escritas no determinado alfabeto de 4 letras: `A`, `C`, `G`, `T`. Essas letras representam as bases nitrogenadas: **A**denina, **C**itosina, **G**uanina e **T**imina.

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 sequências 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` liga-se a base `T` e a base `G` liga-se a base `C`. As bases `A - T` e `C - G` são denominadas pares de **bases complementares**.

No restante do texto consideramos o DNA como uma sequência de letras, onde cada letra representa uma base. Apresentamos na figura 4 um exemplo de representação do DNA como duas sequências de letras, onde cada letra de uma sequência está justaposta a outra. As sequências (fitas) de DNA são postas uma sobre a outra revelando o pareamento ou sobreposição entre as bases nitrogenadas.

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

Figura 4: Exemplo de representação de um pareamento entre duas fitas de DNA.

Um **proteoma** de um organismo é formado pelo conjunto de genes do genoma que codificam proteínas. Nesse trabalho, 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 Figura 5 temos dois exemplos de 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 foi considerada mais adequada para nosso propósito, pois considera a ordem dos genes baseada nas orientações descritas acima.

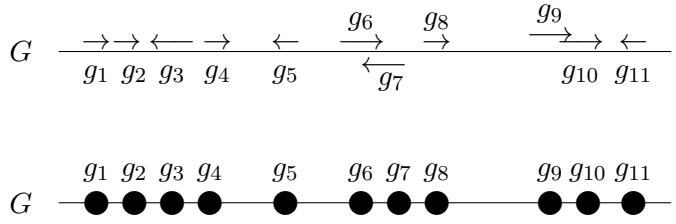

Figura 5: Exemplo de representação dos genes de um proteoma G.

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 '-'. Assim, a ordem dos genes, $g_1, g_2, \ldots, g_n$ é determinada respeitando a relação $P_1 \leq P_2 \leq \ldots \leq P_n$;

Ao analisarmos genomas tentamos predizer quais são os seus genes que podem codificar uma ou mais proteínas. Dessa forma, utilizamos o termo **proteína predita** de um determinado gene para referenciar essas proteínas preditas.

Quando genes evoluem a partir de um determinado gene ancestral em comum, dissemos que esses genes são **homólogos**. Fundamentalmente, existem dois tipos de genes homólogos: **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. Por outro lado, quando os genes g e h descendem de um mesmo gene ancestral, porém pertencem a uma mesma espécie, g e h são denominados **parálogos**.

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;

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 particularmente rica em genes específicos. Apresentaremos posteriormente a quantificação para uma região ser considerada particularmente rica em genes específicos;

Definimos uma **região ortóloga** (**RO**) entre dois proteomas G e H como um um par de regiões (R_i, R_j) onde: R_i e R_j são RGC em G e H respectivamente; R_i e R_j são descendentes de uma mesma região ancestral; e R_i e R_j contêm aproximadamente o mesmo número de genes.

Uma descrição mais formal e detalhada de região ortóloga é apresentada na subseção 4.2.3. A Figura 6 ilustra um exemplo de uma RO.

A clara distinção entre genes ortólogos e parálogos é fundamental para a construção de uma classificação evolucionária dos genes e para uma atribuição funcional confiável das proteínas de novos genomas sequenciados [15].

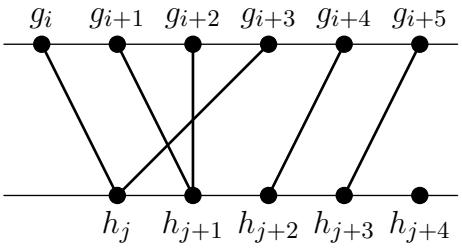

Figura 6: Representação de uma RO.

Estudos sobre a conformação, função e evolução das proteínas têm também revelado a importância de uma unidade de organização presente nas proteínas. Essa unidade é o **domínio** de uma proteína, uma sub-estrutura produzida por qualquer parte de uma cadeia de peptídeos que pode-se "enovelar" independentemente em uma estrutura estável e compacta. Geralmente, um domínio contém entre 40 e 350 aminoácidos. Os diferentes domínios de uma proteína estão frequentemente associados com diferentes funções [15].

Na subseção seguinte apresentamos uma formalização simples para alguns termos da Biologia Molecular e também descreveremos alguns conceitos básicos computacionais, com o objetivo de prover informações suficientes para a compreensão das metodologias descritas no decorrer do texto.

2.2 Definições formais para termos da Biologia Molecular

Nessa seção faremos uma definição mais formal para os termos descritos na seção 2.1 e para os termos de grafos e árvores. Essas definições foram utilizadas no decorrer desse trabalho para podermos descrever mais precisamente as consultas ao banco de dados, que foram construídas a partir da definição das estruturas organizacionais entre dois genomas.

Definição 2.2.1 (Genomas) *Seja uma estrutura G_i o í-ésimo genoma de um conjunto de genomas $S_G = \{G_1, G_2, \ldots\}$. A relação entre dois genomas G_i e G_j é obtida de acordo com a métrica de distância taxonômica definida em 2.2.7.*

Definição 2.2.2 (Genes) *Seja g um gene definido como uma cadeia de caracteres de um dado alfabeto $\sum$. Cada gene está relacionado à exatamente um determinado genoma G. Nesse caso, um determinado gene g_i pode ser igual a um outro genoma g_j.*

Definição 2.2.3 (Hits) *Seja $h_{i,j}$ um* hit *definido como uma função $y = h_{i,j} = h(i,j)$ entre os genes g_i e h_j. O valor $y \in \mathbb{N}^*$ representa um valor ou uma pontuação para a similaridade entre esses determinados genes. Se não for possível determinar a similaridade entre os genes g_i e h_j, ou mesmo, se essa similaridade não existir, então temos que $h(i,j) = 0$.*

O projeto de comparação de genomas GCP [1] utilizou o programa de comparação de sequências SSEARCH [2; 3] para realizar a comparação entre as sequências dos genomas. Este programa é uma uma implementação do algoritmo de Smith-Waterman [20], de código aberto e disponível em [3], e que encontra o melhor alinhamento local entre pares

de sequências. Desta forma, somente os resultados das comparações entre os genes ou proteínas g_i com h_j estão disponíveis, ou seja, assumimos que $h(i,j) = h(j,i)$. Ao assumir que $h(i,j) = h(j,i)$ tivemos que fazer algumas alterações na metodologia adotada. Essas alterações serão descritas na subseção 4.1.

Definição 2.2.4 (Domínio) *Seja D um domínio definido como uma subcadeia de caracteres de um determinado gene g. Um gene g pode ter um ou mais domínios.*

Definição 2.2.5 (Grafo) *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 e é dito **ligar** u a v. Os vértices u e v são denominados **extremos** de e. Na Figura 7, apresentamos um exemplo de 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 $E = \{e_1, e_2, e_3, e_4, e_5\}$.*

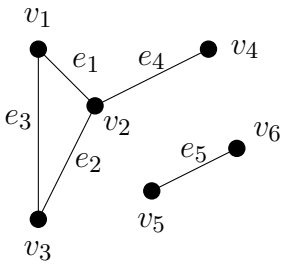

Figura 7: Exemplo de um grafo com seis vértices.

Definição 2.2.6 (Árvore) *Seja uma árvore T definida como um grafo com uma raiz. A partir de cada nó x até a raiz, existe um e somente um único caminho. O tamanho a partir de um nó x até a raiz é denominado de altura ou profundidade do nó. O nó y próximo a x em direção à raiz da árvore, é denominado de nó "pai" de x. Consequentemente, o nó x é denominado de nó "filho" de y.*

Na figura 8, temos uma representação de uma árvore.

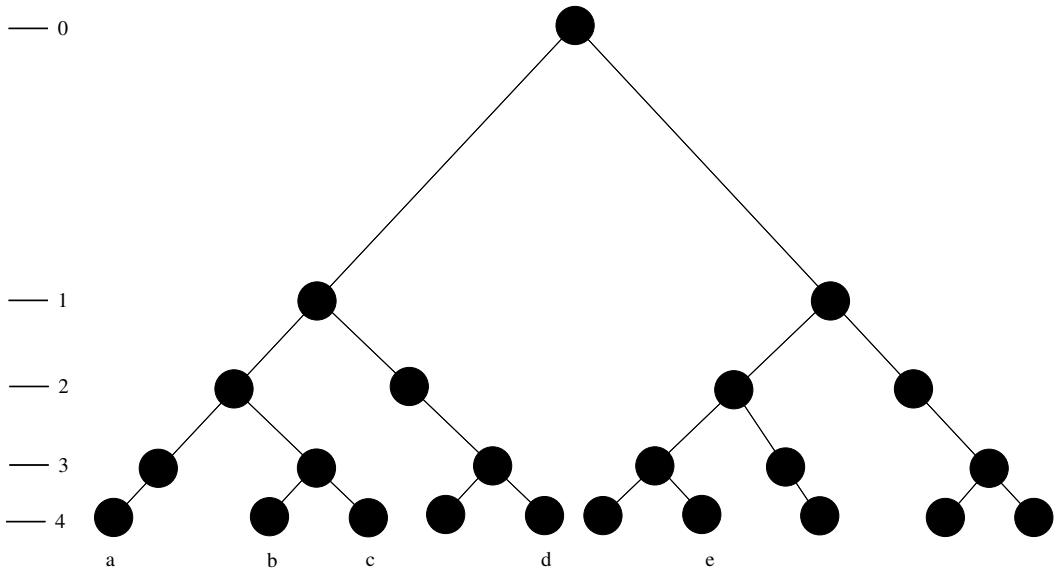

Figura 8: Exemplo de uma árvore. Círculos em preto representam os nós. As letras indicam os genes ou os genomas. Os números do lado esquerdo da figura representam a classificação hierárquica dos genes ou genomas na árvore.

Definição 2.2.7 (Taxonomia) *Seja* $dist_{taxo}(x, y)$ *ou* $||x, y||_{taxo}$ *a medida de distância taxonômica entre dois genomas ou dois genes ao utilizar uma árvore nesse cálculo. O valor de* $dist_{taxo}$ *representa quantos níveis na árvore taxonômica devem ser alcançados para se encontrar o primeiro ancestral comum entre* x *e* y*. Segue que,* $dist_{taxo}(G_i, G_i) = 0$*.*

A definição de taxonomia que descrevemos anteriormente foi baseada na hipótese de que todas as espécies derivam ou são originadas a partir de um ancestral em comum. Dessa forma, podemos utilizar essa informação para "ordenar" os genomas. Neste trabalho estamos utilizando a informação taxonômica extraída do NCBI [5]. Essa classificação mostra o número de nós pais referente ao primeiro ancenstral comum entre dois genes ou genomas. Como exemplo, a figura 8 ilustra uma possível árvore taxonômica. Nesse exemplo, temos as seguintes medidas taxonômicas: $dist_{taxo}(c, b) = 1$, $dit_{taxo}(c, d) = 3$, $dist_{taxo}(a, b) = dist_{taxo}(b, a) = 2$, etc.

3 Trabalhos Relacionados

Viana [18] descreve alguns trabalhos relacionados a comparação de genomas. Dentre os trabalhos descritos, citamos em específico o trabalho apresentado por Kellis e colegas [21], os quais apresentaram um método para a determinação automática da correspondência genômica entre dois genomas. O método apresentado foi utilizado para o alinhamento automático 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 conhecidos genes codantes dessas proteínas.

No trabalho de Kellis e colegas, o algoritmo modela as similaridades entre os genes de dois genomas como um grafo bipartido, com "peso" nas arestas, conectando genes entre duas espécies. Para pontuar as arestas que conectam dois genes em específico são utilizadas ambos, a similaridade entre os aminoácidos das sequências entre dois genes e o tamanho total de alinhamento entre esses genes.

Em resumo, a metodologia descrita por Kellis [21] difere também em relação a metodologia empregada neste trabalho, além também do trabalho de Viana [18]. Em Viana, são utilizados as estruturas dos BBHs para a caracterização dos relacionamentos par-a-par entre os genes de dois genomas. Em Kellis, é utilizada a estrutura de BUS: subconjuntos de genes ótimos locais, tais que todos os *matches* entre os genes então dentro dessa estrutura, além de não ter nenhum gene fora dela. Neste presente trabalho, não temos a figura dos BBHs como estrutura para a caracterização das regiões ortólogas. Neste caso, como estamos utilizando uma ferramenta de comparação de sequências exata, não temos a representação da reciprocidade entre os genes, a qual seria desnecessária tendo em vista o algoritmo utilizado na comparação. Assim, utilizamos apenas os *matches* (*hits* que passaram pelos testes de valores limítrofes) como estruturas básicas para a determinação dos RUNs, das regiões ortólogas, além das regiões específicas entre dois genomas.

Fulton e colegas [22] apresentam um método computacional de alta performance denominado por *Ortholuge* que, após avaliar possíveis genes previamente preditos como ortólogos, identifica quais genes ortólogos refletem mais precisamente a divergência das espécies comparadas, e assim podem mais provavelmente possuírem funções similares. O método *Ortholuge* analise as taxas de distância filogenética envolvendo as duas espécies comparadas e um grupo externo de espécies, observando casos onde a divergência relativa a determinado gene é atípica. Este método também identifica alguns casos de duplicação de genes. Os autores ainda citam que um dos problemas encontrados pelas ferramentas e métodos de comparação ou predição de genes ortólogos, é a dificuldade desses métodos serem facilmente automatizados, para análises de grandes volumes de genomas, sendo assim, estas análises fazem utilização da abordagem dos melhores *hits* recíprocos do BLAST (RBH). Ressaltam, que esses RBH provavelmente irão predizer incorretamente um gene parálogo como um gene ortólogo, quando tivermos trabalhando com sequências genômicas incompletas. Fulton e colegas terminam concluindo que o método tende a melhorar significativamente a especificidade da predição de alto desempenho de genes ortólogos tanto, das espécies de bactérias, quanto de eucariotos.

Wall e Deluca [23] apresentam um método para reconstruir com acurácia os relacionamentos entre genes em diferentes organismos, ou seja, encontrar os genes ortólogos entre eles, tanto quanto, utilizar com acurácia alguma media de diversificação evolucionária. Neste trabalho, apresentam a abordagem denominada por *reciprocal smallest distance algorithm (RSD)*, que segundo os autores, melhora o procedimento comum de obter os hits

RBH, na identificação dos genes ortólogos, pela utilização de um alinhamento global de sequências e também pela utilização de uma estimativa de probabilidade máxima de distâncias evolucionárias, a fim de detectar genes ortólogos entre dois genomas. Ainda segundo os autores, o método RSD encontra tantos possíveis genes ortólogos não reportados pelo procedimento "comum" RBH, pois o método RSD é menos suscetível a enganos na presença de genes parálogos em genomas próximos evolutivamente [23].

4 Estruturas Organizacionais

Nesta seção descrevemos a estratégia que adotamos para realizar a definição e implementação de estruturas organizacionais no banco de dados do PWD-V2. Essas estruturas podem possibilitar encontrar regiões comuns entre os genomas, em relação a genes conservados, determinar os pares de genes ortólogos e as regiões ortólogas, os genes únicos (específicos) e as regiões específicas.Temos por objetivo apresentar a metodologia e implementação das estruturas organizacionais, as quais possibilitem criar consultas, procedimentos ou funções no banco de dados, para que seus resultados possam tentar ajudar a explicar como a reordenação e o reagrupamento de genes podem influenciar nas diferenças entre as funcionalidades dos genomas.

Nas subseções seguintes descreveremos a metodologia utilizada, além de abordarmos a forma de implementação das estruturas organizacionais e de algumas consultas mais simples.

4.1 Metodologia

Descreveremos nessa subseção a metodologia utilizada para definir e implementar as consultas mais simples e as estruturas organizacionais que podem possibilitar a comparação entre os proteomas dos genomas presentes no banco de dados do PWD-V2. Descrevemos a nossa metodologia com base na metodologia apresentada por Viana[18]. Em específico, fizemos uma modificação na metodologia de Viana para determinar os genes específicos e ortólogos, e também para construir as regiões ortólogas e específicas. Decidimos utilizar a metodologia descrita em Viana, pois além de ser uma extensão ao trabalho de Almeida [19], já foi utilizada com sucesso em alguns trabalhos da literatura [24; 25].

Na metodologia temos as descrições dos seguintes termos para nos auxiliar a encontrarmos os genes específicos e ortólogos:

1. Considere g e h genes dos proteomas G e H respectivamente;
2. Considere $s(g, h)$ uma medida estatística de similaridade entre os genes g e h, de tal forma que, quanto menor $s(g, h)$, mais similares serão os genes g e h.
3. Considere A o alinhamento entre as sequências dos genes g e h. Sejam I_g, J_g, I_h, J_h posições dos genes g e h conforme definição abaixo:
   - I_g e J_g são, respectivamente, o primeiro e o último símbolos do gene g que aparecem no alinhamento A; e
   - I_h e J_h são, respectivamente, o primeiro e o último símbolos do gene h que aparecem em A.

   A **cobertura** de um alinhamento é calculada a partir da porcentagem do tamanho de um gene em relação ao seu alinhamento. Neste caso, a cobertura do alinhamento A em g , denotada por $c(A, g)$, é dada pelo percentual de $|g|$ que aparece em A. Desta forma, temos:

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

Esta definição vale também para a cobertura de A em relação ao gene h, ou seja, para $c(A, h)$ temos:

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

4.1.1 Genes Específicos e Ortólogos

Nesta seção descreveremos os critérios utilizados para a determinação dos genes ortólogos e específicos, baseado nas definições apresentadas anteriormente. Neste trabalho utilizamos os seguinte critério para a determinação dos genes ortólogos e também os genes específicos:

- 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 $c(A, g) \leq P$ e $c(A, h) \leq P$, onde P é um limite fixo.

  Na definição da metodologia proposta por Viana [18], tínhamos a visão de genes **fortemente ortólogos**, que eram os genes g e h, ortólogos entre sei, e também o gene h era o gene de H que possuía a menor medida estatística de similaridade com o gene g, para todos os genes ortólogos $h^{'}$ de H, e vice-versa. Neste trabalho, não temos a comparação entre o gene h de H com o gene g de G nesta ordem, assim, tivemos que ajustar a metodologia para utilizarmos somente como gene "fortemente ortólogo" aquele gene h que possuir a menor medida estatística de similaridade com um gene g, ou seja, não exigindo que a comparação entre h e g, nesta ordem, seja feita para ser utilizada na metodologia. Conforme apresentaremos na seção de implementação 4.2, esta definição estará de acordo com a escolha da ferramenta (SSEARCH) utilizada para implementar as comparações entre dois proteomas G e H.

- 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 4.2.2, descreveremos como essa comparação e como esses valores serão obtidos.

4.1.2 Regiões específicas (REs)

O problema de determinar REs pode ser modelado para o problema computacional conhecido como *subcadeia de máxima soma*, conforme definido no trabalho de Viana [18]. Naquele trabalho temos a citação de algumas aplicações para esse problema, as quais poderiam ser a identificação de regiões de transmembranas e domínios de ligação, ambos também referenciados no trabalho de Ruzzo e Tompa [26].

Assim como no trabalho de Almeida [19] e Viana [18], utilizamos a mesma estratégia para determinar as estruturas organizacionais, a qual consiste em atribuir determinados valores as proteínas do proteoma de um genoma, onde um valor δ é atribuído para as

proteínas não específicas e um valor Δ para as proteínas específicas, de tal forma que $\Delta > \delta$. Assim, a seqüência de entrada para o problema é constituída pelos valores atribuídos as proteínas.

A implementação dessa estratégia utiliza um algoritmo para encontrar todas as subcadeias maximais. Assim como em Viana, utilizamos a versão $O(n)$ do algoritmo descrito por Cáceres e colegas [27], que resolve eficientemente esse problema.

4.1.3 Regiões Ortólogas (ROs)

Nesta seção descreveremos a metodologia para encontrar as regiões ortólogas entre dois proteomas. Em específico, utilizaremos a descrição da metodologia apresentada por Almeida [19] e Viana [18]. Apresentamos as definições encontradas nesses trabalhos para possibilitar uma melhor compreensão das metodologias apresentadas pelos autores:

Definição 4.1.1 (RUN) *Sejam dois proteomas G e H. Seja α uma RGC de G formada pelas proteínas $g_i, \ldots, g_k$ e β uma RGC de H formada pelas proteínas $h_j, \ldots, h_l$, tais que $k - i + 1 = l - j + 1$, $k > i$ e $l > j$. Definimos que α e β formam um **RUN** se quaisquer uma das seguintes sequências de pares de proteínas ortólogas ocorrerem:*

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

No trabalho de Almeida temos a classificação dos RUNs como paralelo ou anti-paralelo. Neste caso, Almeida define um RUN como **paralelo** quando a sequência de pares ortólogos correspondem a opção número 1 anteriormente. Por outro lado, quando a sequência de pares de ortólogos correspondem a opção número 2, Almeida classifica o RUN como **anti-paralelo**. A estrutura de *RUN* é utilizada para permitir a construção de regiões ortólogas. Almeida também classifica os RUNs como consistentes ou inconsistentes. Neste trabalho não utilizamos essa classificação para a construção das regiões ortólogas.

Após definir o conceito de RUN, podemos rever a definição do conceito de RO apresentado por Almeida e Viana:

Definição 4.1.2 (Região Ortóloga) *Uma região ortóloga R pode ser definida 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] dos runs não seja maior que um determinado valor fixo k, em número de genes.*

Conforme a estratégia apresentada em Viana, a construção das ROs consiste em percorrer todos os RUNs, da esquerda para a direita (conforme a ordem dos genes de um proteoma), e fazer a junção daqueles RUNs que atendem ao critério de distância (2) entre RUNs. Almeida [19] cita os valores mais adequados para os parâmetros M e k para a junção de RUNs entre genomas de procariotos.

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

4.2 Implementação das Estruturas

Nesta subseção abordaremos a implementação da metodologia apresentada anteriormente. Esta implementação foi descrita baseada no esquema e na quantidade de dados que tivemos que reduzir do espaço amostral dos dados gerados pelo GCP, a fim de possibilitar que as consultas fossem executadas e validadas antes de serem executadas para todo o registros dos resultados das comparações entre os genomas envolvidos.

4.2.1 Requisitos das Estruturas Organizacionais no ProteinWorlDB

Nesta seção abordaremos alguns requisitos necessários ao ProteinWorlDB para a implementação das estruturas organizacionais descritas na seção 4.1. Entre os requisitos, podemos citar a necessidade de utilização de um subconjunto do conjunto de dados de resultados e a necessidade de uma sintonia do ProteinWorlDB para que consigamos responder as consultas necessárias à construção das Estruturas Organizacionais.

Embora estejamos utilizando um subconjunto dos dados total, este ainda é significativo em quantidade de dados (15 GB), para podermos fazer a prova de conceito da implementação das estruturas organizacionais, frente a quantidade total de dados de resultados passar de 300 GB de dados comprimidos. Desta forma, tivemos que aplicar algumas técnicas de bancos de dados para fazer a sintonia do projeto físico, com a finalidade de atender aos objetivos que desejamos alcançar. Como sabemos, um projeto de um banco de dados deve ser dirigido pelas necessidades de processamento, assim como pelas necessidades de gerenciamento dos dados. Conforme as necessidades de processamento mudem, o projeto do banco de dados precisa responder a esse dinamismo, fazendo com que as mudanças necessárias sejam feitas no esquema conceitual e, se necessárias, refletidas no esquema lógico e físico. Dentre a natureza das mudanças possíveis no projeto de um banco de dados [28] temos o **particionamento horizontal** e a criação de algumas estruturas de acesso, como **visões materializadas**.

Neste trabalho, assim como na literatura, temos consultas que não são executadas muito rapidamente, devido a quantidade de tabelas e registros envolvidos na obtenção dos resultados. Muitas vezes, tentamos utilizar as técnicas possíveis de otimização de consultas e acabamos descobrimos que não é possível executar as consultas que desejamos, de uma forma mais eficiente possível, sem termos simplesmente que reestruturar completamente os nossos dados. O que acabamos fazendo muitas vezes é armazenar dados pré-consultados, para que não tenhamos que executar as mesmas consultas novamente, quando precisarmos dos dados. Este procedimento é denominado por "*caching* " fora do mundo de banco de dados. Na verdade, o que estamos fazendo é criar uma visão materializada dos dados, ou seja, criando um outro método de acesso para os dados que necessitamos e, além disso, estamos transformando-os em uma tabela do banco de dados, a qual detém os dados resultantes de uma consulta, ao invés de simplesmente termos apenas um visão dos dados a serem consultados.

No particionamento horizontal temos fatias de uma tabela armazenadas como tabelas distintas. Cada tabela possui o mesmo conjunto de atributos mas contém conjunto de registros distintos. Desta forma, podem ser feitas consultas específicas para uma faixa de valores, onde apenas algumas tabelas podem ser consultadas. Decidimos utilizar esta técnica de banco de dados para repartir o volume de dados de resultados em diversas tabelas, com a finalidade de melhorar o desempenho das consultas. Descrevemos o particionamento

horizontal dos dados de resultado do projeto GCP na seção 4.2.1

Visão Materializada

Nesta seção descreveremos como fizemos para implementar visão materializada no PostgreSQL. Atualmente visões materializadas não estão disponíveis como um método de acesso nativo do PostgreSQL versão 8.4. No entanto, são certamente possíveis de serem implementadas no PostgreSQL, devido as possibilidades da linguagem PL/pgSQL e do sistema de *triggers*.

Como sabemos, uma visão materializada pode ser vista como uma tabela que realmente contém linhas, mas se comporta como uma visão, ou seja, os dados na tabela mudam quando os dados das tabelas de referência são alterados. Existem alguns tipos de visão materializada, com relação a atualização dos dados:

Snapshot: são visões materializadas que são atualizadas manualmente, e por isso, são mais simples de serem implementadas.

Eager: são visões materializadas que são atualizadas tão logo qualquer mudança seja realizada na base de dados.

Lazy: são visões materializadas que são atualizadas somente quando a transação efetuar um *commit*.

Very Lazy: são visões materializadas que são funcionalmente equivalentes as visões materializadas **Snapshot**. A diferença consiste em que as mudanças são registradas de forma incremental e aplicadas quando a tabela é atualizada manualmente.

Neste trabalho utilizamos a implementação de visões materializadas descritas por Gardner [29]. Mais especificamente, utilizamos a implementação de visões materializadas do tipo **Snapshot**, por serem mais fácil de implementar e também por analisarmos que as atualizações nesta base não serão muito frequentes, e desta forma, poderão ser realizadas manualmente após a atualização de determinadas tabelas.

A proposta de implementação de visões materializadas descrita por Gardner [29] para o PostgreSQL versão 8.4 é baseada na criação de uma tabela auxiliar (*matviews*) para armazenar os dados sobre as visões materializadas, na criação de funções para criar (*create_ matview*), apagar (*drop_ matview*) e atualizar (*refresh_ matview*) visões materializadas. Maiores detalhes sobre a forma de criação de visões materializadas do tipo **Snapshot** podem ser vistas em [29].

Para utilizarmos a implementação padrão da metodologia de criação das estruturas organizacionais, descritas por Nalvo [19] e estendidas em Viana [18], sobre os dados do banco de dados do ProteinWorlDB, foi necessário alterarmos a forma como as principais estruturas de dados implementadas por Viana fossem atualizadas. Para isso, foi necessário fazermos consultas diretamente a base de dados do ProteinWorlDB e capturar os dados necessários ao preenchimento das estruturas relativas aos dados de genomas em comparação (*tOrf*), assim como a estrutura (*tHit*) referente aos dados dos resultados das comparações entre as proteínas dos genomas comparados.

Para realizarmos o preenchimento das estruturas de dados referentes aos dados dos genomas comparados, teríamos que consultar as tabelas *genomas, genes e proteínas* respectivamente, com a finalidade de listarmos as proteínas codificadas por um ou mais genes,

que pertencem a um determinado genoma e que foram utilizadas no procedimento de comparação de sequências. Para tentarmos otimizar a listagem dos dados referentes a dois genomas em específico, aos genomas sendo comparados, acabamos criando uma visão materializada dos dados de proteínas (*mv_genome_genes_proteins*) necessários ao preenchimento da estrutura de dados *tOrf* de todos os genomas comparados. Foram criados nessa "visão materializada" também alguns índices sobre os identificadores de genomas, proteínas e genes, para facilitar o acesso a esses dados de forma mais eficiente. A figura 9 ilustra a visão referente a "visão materializada" que foi criada para auxiliar no preenchimento das estruturas de dados necessárias a computação das estruturas organizacionais entre dois genomas.

```
CREATE VIEW "public"."v_genome_genes_proteins" (
    gs_gbkid, gs_gbkdefinition, gs_gi, gs_gbkdivision, gs_length, gs_mol_type, gs_status,
    gs_projectid, gs_seqtype, gs_gc_content, gs_taxonomy_id,
    cds_fiocruzid, cds_region, cds_gs_gbkid, cds_geneid,
    g_geneid, g_transcriptid, g_reg_start, g_reg_stop, g_strand, g_index,  g_gc_content,
    p_fiocruzid, p_gbkid, p_uniprotid, p_length, p_definition, p_taxonomy_id, p_gbkidwversion)
AS
SELECT gs.gbkid AS gs_gbkid, gs.gbkdefinition AS gs_gbkdefinition, gs.gi AS
    gs_gi, gs.gbkdivision AS gs_gbkdivision, gs.length AS gs_length,
    gs.mol_type AS gs_mol_type, gs.status AS gs_status, gs.projectid AS
    gs_projectid, gs.seqtype AS gs_seqtype, gs.gc_content AS gs_gc_content,
    gs.taxonomy_id AS gs_taxonomy_id, cds.fiocruzid AS cds_fiocruzid,
    cds.region AS cds_region, cds.gbkid AS cds_gs_gbkid, cds.geneid AS
    cds_geneid, g.geneid AS g_geneid, g.transcriptid AS g_transcriptid,
    g.reg_start AS g_reg_start, g.reg_stop AS g_reg_stop, g.strand AS g_strand,
    g.index AS g_index, g.gc_content AS g_gc_content, p.fiocruzid AS
    p_fiocruzid, p.gbkid AS p_gbkid, p.uniprotid AS p_uniprotid, p.length AS
    p_length, p.definition AS p_definition, p.taxonomy_id AS p_taxonomy_id,
    p.gbkidwversion AS p_gbkidwversion
FROM (((genomic_sequence gs JOIN cds ON (((gs.gbkid)::text =
    (cds.gbkid)::text))) LEFT JOIN gene g ON ((cds.geneid = g.geneid))) LEFT
    JOIN protein p ON ((cds.fiocruzid = p.fiocruzid)))
ORDER BY gs.gbkid, g.reg_start, g.reg_stop;
```

Figura 9: Visão referente a implementação da "visão materializada" *mv_genome_genes_proteins*. utilizada no preenchimento dos dados referentes as estruturas de dados de genomas.

Particionamento dos dados

Nesta seção descreveremos como fizemos o particionamento dos dados de resultados. Atualmente, o SGBD do PostgreSQL versão >= 8.4 suporta o particionamento através de herança entre as tabelas. Neste caso, cada partição deve ser criada como uma tabela "filha" de uma única tabela "pai". A tabela "pai" existe para representar o conjunto de dados total e normalmente não contém elementos. No SGBD PostgreSQL 8.4 estão implementadas as seguintes formas de particionamento:

Faixas: Nesta forma, a tabela é particionada em "faixas" por uma coluna chave ou por um conjunto de colunas, sem termos sobreposições entre as faixas de valores atribuídas as diferentes partições.

Listagem: Nesta forma, a tabela é particionada pela listagem de valores que devem aparecer em cada partição.

Neste trabalho utilizamos o particionamento por faixas. Como no arquivos de resultados de comparação das sequências de proteínas temos os identificadores das proteínas envolvidas na comparação, estes identificadores foram utilizados como valores para particionamento dos arquivos em várias tabelas, onde cada tabela refere-se a uma faixa de identificadores referentes as sequências de consulta.

Implementando o particionamento

Para realizarmos o particionamento dos dados de resultados, fizemos primeiramente a criação das tabelas de partições e depois importamos os dados dos arquivos de resultados. A tabela que originalmente armazena os dados dos arquivos de resultados é a tabela *hit*. Para criar as partições da tabela *hit* fizemos os seguintes passos:

- Criamos a tabela *hit* que contém todos os atributos referentes as colunas de uma linha do arquivo de resultado. Esta tabela *hit* não contém nenhum dado armazenado, além de não conter quaisquer restrições sobre a tabela.
- Criamos algumas tabelas "filhas" da tabela *hit*. Essas tabelas filhas foram denominadas com o sufixo das faixas ao qual os identificadores das sequências pertencem.
- Acrescentamos restrições as tabelas de partição para permitir a inserção dos valores para cada partição.
- Para cada partição criada, criamos também um índice sobre a coluna de identificação da sequência query (*query_fiocruzid*).
- Definimos uma *trigger* para direcionar a inserção dos dados da tabela *hit* para a partição "filha" apropriada.
- Alterar as consultas aos dados de resultados especificando o parâmetro de configuração de **restrição de exclusão** (*constraint_exclusion*) do PostgreSQL. Com esse parâmetro habilitado as consultas serão otimizadas para buscar apenas nas partições ao qual o predicado é atendido.

Os passos anteriormente descritos foram implementados através da criação do *script Perl* (*protein_buckets.pl*) que faz o particionamento dos identificadores das sequências de proteínas dos genomas utilizados nas comparações. Em nossos experimentos decidimos particionar o conjunto de identificadores das sequências (3.812.663) em 60 partições, com a finalidade de que cada identificador de sequência presente nos arquivos de resultados possam ser distribuídos, de uma tal forma que as tabelas "filhas" não fiquem com muitos dados, atrasando assim a obtenção dos resultados das consultas. Embora utilizemos os identificadores das sequências de proteínas dos genomas para a construção das partições, nem todas esses identificadores estão presentes nos arquivos de resultados, uma vez que apenas aqueles que fizeram *hit* com alguma outra sequência estão representados. A decisão por utilizar os identificadores das sequências dos genomas comparados, ao invés dos identificadores das sequências presentes nos arquivos de resultados foi devido, principalmente, a possibilidade de atualização do conjunto de dados e também para tentar agrupar os resultados de um mesmo genoma em um número menor de tabelas possível, facilitando a sua obtenção.

O *script protein_ buckets* executa o particionamento sobre uma lista de identificadores de sequências de proteínas dos genomas comparados e gera como resultado o seguintes arquivos:

hit_pp_qid.table.sql: Armazena as sentenças SQL para se criar as tabelas e as restrições sobre cada uma delas. A figura 10 ilustra o trecho do arquivos gerado quando estamos utilizando 60 partições dos identificadores de sequência.

```
CREATE TABLE hit_pp_qid (LIKE hit_pp INCLUDING DEFAULTS INCLUDING CONSTRAINTS);
CREATE TABLE hit_pp_qid_ge4501845_l15220314 (CHECK (query_fiocruzid >= 4501845 AND query_fiocruzid < 15220314)) INHERITS (hit_pp_qid);
CREATE TABLE hit_pp_qid_ge15220314_l15803238 (CHECK (query_fiocruzid >= 15220314 AND query_fiocruzid < 15803238)) INHERITS (hit_pp_qid);
CREATE TABLE hit_pp_qid_ge15803238_l16128163 (CHECK (query_fiocruzid >= 15803238 AND query_fiocruzid < 16128163)) INHERITS (hit_pp_qid);
CREATE TABLE hit_pp_qid_ge16128163_l17988013 (CHECK (query_fiocruzid >= 16128163 AND query_fiocruzid < 17988013)) INHERITS (hit_pp_qid);
CREATE TABLE hit_pp_qid_ge17988013_l21244594 (CHECK (query_fiocruzid >= 17988013 AND query_fiocruzid < 21244594)) INHERITS (hit_pp_qid);
...
```

Figura 10: Trecho do arquivo de resultado de criação de tabelas "filhas" para o particionamento dos identificadores de sequências.

hit_pp_qid.index.sql: Armazena as sentenças SQL para se criar os índices sobre cada uma das tabelas. A figura 11 ilustra o trecho do arquivo gerado para a criação dos índices quando utilizamos 60 partições nos identificadores de sequências.

```
CREATE INDEX hit_pp_qid_ge4501845_l15220314_hit_pp_qid_idx ON hit_pp_qid_ge4501845_l15220314 (query_fiocruzid);
CREATE INDEX hit_pp_qid_ge15220314_l15803238_hit_pp_qid_idx ON hit_pp_qid_ge15220314_l15803238 (query_fiocruzid);
CREATE INDEX hit_pp_qid_ge15803238_l16128163_hit_pp_qid_idx ON hit_pp_qid_ge15803238_l16128163 (query_fiocruzid);
CREATE INDEX hit_pp_qid_ge16128163_l17988013_hit_pp_qid_idx ON hit_pp_qid_ge16128163_l17988013 (query_fiocruzid);
CREATE INDEX hit_pp_qid_ge17988013_l21244594_hit_pp_qid_idx ON hit_pp_qid_ge17988013_l21244594 (query_fiocruzid);
...
```

Figura 11: Trecho do arquivo de resultado de criação dos índices sobre as tabelas "filhas" do particionamento dos identificadores de sequências.

hit_pp_qid.trigger.sql: Armazena as sentenças para criar uma função e sua *trigger* com as faixas de valores e as respectivas tabelas que devem ser utilizadas durante o processo de carga dos dados. A figura 12 ilustra o trecho do arquivo gerado para a criação da função e *trigger* quando utilizamos 60 partições nos identificadores de sequências.

```
CREATE OR REPLACE FUNCTION hit_pp_qid_insert_trigger()
RETURNS TRIGGER AS $$
BEGIN
IF ( NEW.query_fiocruzid >= 4501845 AND NEW.query_fiocruzid < 15220314 ) THEN INSERT INTO hit_pp_qid_ge4501845_l15220314 VALUES (NEW.*);
ELSIF ( NEW.query_fiocruzid >= 15220314 AND NEW.query_fiocruzid < 15803238 ) THEN INSERT INTO hit_pp_qid_ge15220314_l15803238 VALUES (NEW.*);
ELSIF ( NEW.query_fiocruzid >= 15803238 AND NEW.query_fiocruzid < 16128163 ) THEN INSERT INTO hit_pp_qid_ge15803238_l16128163 VALUES (NEW.*);
ELSIF ( NEW.query_fiocruzid >= 16128163 AND NEW.query_fiocruzid < 17988013 ) THEN INSERT INTO hit_pp_qid_ge16128163_l17988013 VALUES (NEW.*);
...
ELSE
   RAISE EXCEPTION 'Date out of range.  Fix the hit_pp_qid_insert_trigger() function!';
END IF;
RETURN NULL;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER insert_hit_pp_qid_trigger
BEFORE INSERT ON hit_pp_qid
FOR EACH ROW EXECUTE PROCEDURE hit_pp_qid_insert_trigger();
```

Figura 12: Trecho do arquivo de resultado da criação da função e *trigger* para direcionar a inserção dos dados as partições específicas.

hit_pp_qid.drop.all.sql: Armazena as sentenças SQL para apagar as tabelas, a função e a *trigger* associada. A figura 13 ilustra o trecho do arquivo gerado para a remoção

das tabelas, índices, função e *trigger* quando utilizamos 60 partições dos identificadores de sequências.

```
DROP TABLE IF EXISTS hit_pp_qid_ge4501845_l15220314;
DROP TABLE IF EXISTS hit_pp_qid_ge15220314_l15803238;
DROP TABLE IF EXISTS hit_pp_qid_ge15803238_l16128163;
DROP TABLE IF EXISTS hit_pp_qid_ge16128163_l17988013;
DROP TABLE IF EXISTS hit_pp_qid_ge17988013_l21244594;
...
DROP TRIGGER IF EXISTS insert_hit_pp_qid_trigger ON hit_pp_qid;
DROP FUNCTION IF EXISTS hit_pp_qid_insert_trigger();

DROP TABLE IF EXISTS hit_pp_qid;
```

Figura 13: Trecho do arquivo de resultado da remoção das tabelas, índices, função e *trigger*.

Desenvolvemos o *script protein_ buckets* com a finalidade de podermos variar a quantidade de partições sem termos que fazê-las de forma manual. Assim, podemos incrementalmente adicionar mais arquivos de resultados de comparações ao banco de dados, e ir verificando a quantidade de registros em cada partição. Desta forma, podemos avaliar a distribuição dos registros a fim de tentar fazer uma distribuição mais homogênea dos registros de resultados entre as partições, possibilitando avaliar também o desempenhos das consultas para o preenchimento das estruturas de dados necessárias à construção das estruturas organizacionais entre dois genomas.

4.2.2 Determinação dos genes específicos e ortólogos.

Nesta seção abordaremos como implementamos a forma para determinarmos os genes específicos e ortólgos entre dois genomas. Diferentemente do trabalho apresentado por Almeida [19] e Viana [18], no presente trabalho não precisamos realizar o procedimento de comparação dos genes/proteínas entre dois genomas, pois este processo já foi realizado no projeto GCP, e gerou uma quantidade de resultados da ordem de 300 GB de dados. No entanto, para implementar a medida de similaridade descrita na metodologia 4.1.1, utilizaremos a medida de significância estatística $E - value$ presente nos arquivos de resultados.

Considerarmos um gene g específico em relação ao proteoma H, se g não obteve *hits* com valor de $E - value <= T_e$, onde T_e é um valor de $E - value$ limite. Este valor inicialmente foi definido como $S^{'} = 10^{-3}$, mas é um parâmetro fornecido pelo usuário, e desta forma pode ser alterado.

Para determinarmos os genes ortólogos, é necessário primeiramente especificarmos um relacionamento (*match*) entre os genes g_i e h_j dos genomas G e H, respectivamente. Desta forma, podemos definir que a implementação de um **match** entre os genes g_i e h_j ocorre quando um determinado gene g_i encontrar o gene h_j como *hit*, com determinados valores limites $S = 10^{-5}$ e $P = 60$. Os valores de S e P também são parâmetros fornecidos pelos usuários. Diferente do trabalho de Viana [18], não é necessário termos a comparação do gene h_j com o gene g_i, nesta ordem, para definirmos o relacionamento entre eles como um *match*, devido ao programa de comparação de sequências que foi utilizado para gerar os resultados das comparações. Em específico, como estamos utilizando o programa SSEARCH [2; 3] e este programa é uma implementação rigorosa do algoritmo local de programação dinâmica de Smith-Waterman [4], só temos no conjunto de dados de resultados as comparações entre os genes g_i de um genoma G e h_j de um genoma H, nesta ordem.

Com a especificação dos genes g_i e h_j que fizeram *match* entre dois genomas G e H respectivamente, temos a determinação dos pares de genes ortólogos e também os genes específicos. Posteriormente a determinação desses tipos de genes, podemos utilizá-los para a construção das estruturas organizacionais das regiões ortólogas e específicas.

4.2.3 Determinação das estruturas organizacionais.

Nessa fase descreveremos a implementação das estruturas organizacionais das regiões específicas, os runs e as regiões ortólogas conforme descritas a partir da Seção 4.1. A proposta de implementação das estruturas organizacionais deste trabalho consiste na modificação do preenchimento das estruturas de dados do trabalho apresentado por Viana [18], para que estas possam ser atualizadas através dos dados do banco de dados do ProteinWorldDB e, desta forma, as estruturas geradas possam ser posteriormente armazenadas no próprio banco de dados do ProteinWorlDB. Apresentamos nas seções seguintes as implementações das estruturas organizacionais propostas no trabalho de Almeida [19] e Viana [18], com as modificações necessárias tanto por devido ao programa de comparação de sequências utilizado, quanto ao preenchimento das estruturas de dados necessárias a construção das estruturas organizacionais entre dois genomas.

Regiões Específicas

A implementação das regiões específicas é realizada conforme a estratégia descrita na Seção 4.2.3, que também foi descrita no trabalho de Viana [18]. 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 subsequências contíguas de máxima soma, de uma sequência de entrada A composta pelos valores 1 e -1.

Neste trabalho, continuamos a utilizar o algoritmo Subseqüências-Maximais descrito Cáceres e colegas [27] para construir todas as subsequências contíguas de máxima soma. Neste caso, a modificação referente ao trabalho de Viana consiste apenas no preenchimento da estrutura de dados de proteínas e sua lista de proteínas similares utilizadas nas comparações, as quais são utilizados para determinar o vetor de 1 e -1 passado como argumento do método *maximalSubSeqsO_N*. No trabalho de Viana [18] temos a descrição do pseudo-código referente a implementação do algoritmo para encontrar as subsequências de máxima soma.

Runs

A implementação dos RUNs está descrita conforme apresentado na metodologia 4.1. Neste trabalho em específico, não utilizamos a definição de *hits bidirecionais* (genes fortemente ortólogos) para realizar a implementação dos RUNs. Neste caso, estamos considerando que um *match* é apenas um gene que possui um *hit* com outro gene presente nos arquivos de resultados, e que passou pelos valores limites estabelecidos.

Assim como a definição apresentada por Viana, temos que um RUN é uma sequência 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 diagonais consecutivas, onde as posições estão preenchidas com 1. No trabalho apresentado por Viana [18] temos a descrição do algoritmo CONSTRÓI-RUN que possibilita determinar os RUNs entre dois genomas. A Figura 14 ilustra graficamente um exemplo de um RUN encontrado entre os genomas da *Xanthomonas axonopodis pv. citri str. 306 e Xanthomonas campestris pv. campestris str. ATCC 33913*. Nesta figura podemos observar a orientação das setas à esquerda, representando os genes da fita '-', enquanto que os genes pertencentes a fita '+' são representados pelas setas para a direita.

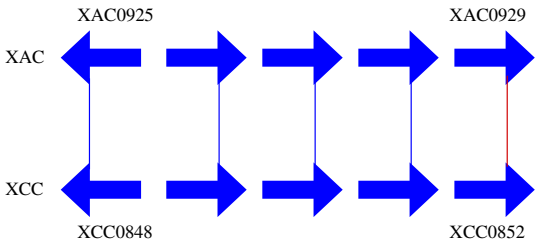

Figura 14: Representação gráfica de um RUN.

Regiões ortólogas

Conforme a descrição da metodologia apresentada por Almeida [19] e Viana [18], temos que uma região ortóloga pode ser composta por M pares de genes/proteínas ortólogos ou pela união de RUNs com pelo menos M pares de ortólogos, que possuem uma distância de no máximo um determinado valor fixo k.

Utilizamos a mesma implementação apresentada por Viana [18] para a noção de distância que permita a união de RUNs próximos. Segunda a implementação temos:
Sejam os RUNs R_1 e R_2 entre os genomas G e H. Pela definição de RUN 4.1.1 podemos representar R_1 e R_2 como:

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

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: O valor do tamanho máximo do menor intervalo entre os RUNs;
- max_large_gaps: O Valor do tamanha máximo do maior intervalo entre os RUNs.

Assim como em Viana, juntaremos os RUNs conforme a seguinte regra de distância:

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

Os RUNs que obedecem a relação anterior são denominados por RUNs **próximos** e o procedimento de união dos RUNs de **junção**. Utilizamos o algoritmo incremental descrito por Almeida [19] para determinar as Regiões Ortólogas entre dois proteomas. Diferentemente do trabalho de Almeida e Viana, não temos a utilização dos BBHs como a definição de um RUN, permitindo assim a sua junção aos demais RUNs, caso seja possível. Desta forma, as regiões Ortólogas são formadas exclusivamente apenas por RUNs próprios ou pela junção de RUNs próximos. No trabalho de Viana e Almeida, poderíamos ter uma situação na qual tivéssemos um RUN próximo a um *match* isolado, que poderiam não ser juntados se estiverem isolados no decorrer do proteoma. No entanto, esse *match* poderia ser unido ao RUN, formando assim uma região com 3 ou mais *matches*. E nesse caso, o *match* só seria utilizado se este contribuísse significativamente para a região, ou seja, se fosse um BBH e também obedecesse a regra de distância entre RUNs. Neste caso, o BBH era considerado como um RUN. Como não tempos BBHs em nosso trabalho, não teremos essa outra possibilidade de junção entre RUNs. A figura 15 ilustra graficamente uma região ortóloga resultante da junção de 3 RUNs com 2, 3 e 2 *matches* respectivamente, entre os organismos *Xylella fastidiosa 9a5c* e *Neisseria meningitidis MC58*.

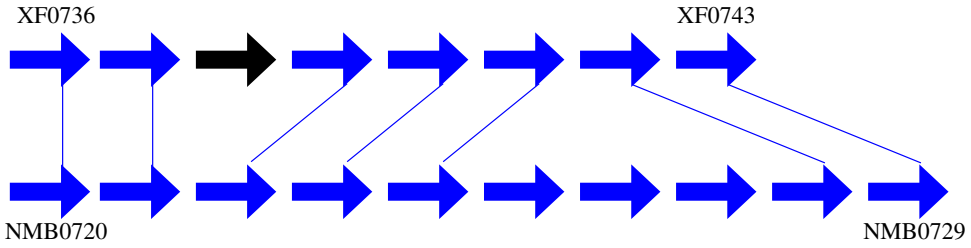

Figura 15: Representação gráfica de uma Região Ortóloga. O gene de cor preta ilustra um gene anotado como hipotético. O sentido das setas representam as orientações dos genes.

5 Outras Consultas

Nesta seção iremos descrever outras possíveis consultas que podem ser executadas na base de dados com a finalidade de responder outras questões relacionadas a quantificação, anotação e seleção de determinados dados específicos do banco de dados, como aquelas relacionadas em 1. Embora não seja o objetivo direto deste trabalho, acreditamos que estas possam ser úteis para a construção de outras análises mais complexas.

Questões estatísticas

- Quantos genomas e genes existem no banco de dados?

```
SELECT COUNT(*)
FROM genomic_sequence;
```

- Quantos e quais são os genes pertencentes a uma determinada fita de cada genoma?

```
SELECT gs.gbkid, gs.gbkdefinition, g.strand, COUNT(g.geneid)
FROM   genomic_sequence gs, cds, gene g
WHERE g.geneid = cds.geneid AND
      cds.gbkid = gs.gbkid
GROUP BY gs.gbkid, gs.gbkdefinition, g.strand
```

Resposta limitada a 20 registros:

```
gbkid               gbkdefinition                          strand count
NC_000117.1 Chlamydia trachomatis D/UW-3/CX, complete genome. + 441
NC_000117.1 Chlamydia trachomatis D/UW-3/CX, complete genome. -454
NC_000521.3 Plasmodium falciparum 3D7 chromosome 3. + 126
NC_000521.3 Plasmodium falciparum 3D7 chromosome 3. -115
NC_000834.1 Branchiostoma floridae mitochondrion, complete genome. + 12
NC_000834.1 Branchiostoma floridae mitochondrion, complete genome. -1
NC_000843.1 Neurospora intermedia mitochondrial plasmid Harbin-3, complete sequence. + 1
NC_000843.1 Neurospora intermedia mitochondrial plasmid Harbin-3, complete sequence. -1
NC_000844.1 Daphnia pulex mitochondrion, complete genome. + 9
NC_000844.1 Daphnia pulex mitochondrion, complete genome. -4
```

- Quantos genomas pertencem a uma determinada classificação taxonômica?

```
SELECT COUNT(gbkid)
FROM genomic_sequence gs, taxonomy t
WHERE gs.taxonomy_id = t.taxonomy_id AND
      t.name LIKE '%bacteria%';
```

Questões sobre seleção de determinados dados

- Quais são todas as proteínas para um determinado *Enzyme Commision number - EC ec_id*?

```
SELECT ea.fiocruzid, p.definition
FROM enzyme_annotation ea
JOIN protein p ON ea.fiocruzid = p.fiocruzid
WHERE ea.ec_id = '4.4.1.14'
```

- Quais são todos os hits para um determinado gene g_i com $E - value \leq threshold$?

```
SELECT p.fiocruzid, p.definition, h.e_value, h.bit_score, h.sw_score
FROM hit_pp_qid h
JOIN protein p ON h.subject_fiocruzid = p.fiocruzid
WHERE h.query_fiocruzid = 10957467 AND
    h.e_value <= 1.0e-5;
```

Resposta limitada a 9 registros:

```
fiocruzid definition e_value bit_score sw_score
13474488 ABC transporter ATP-binding protein [Mesorhizobium loti MAFF303099]. 1,4E-20 93 407
13475670 ABC transporter ATP-binding protein [Mesorhizobium loti MAFF303099]. 5E-20 90,9 397
13475228 ABC transporter ATP-binding component [Mesorhizobium loti MAFF303099]. 2,2E-19 89,3 390
13474015 ABC transporter, ATP-binding protein [Mesorhizobium loti MAFF303099]. 3,6E-19 88,5 386
13474418 ABC transporter ATP-binding protein [Mesorhizobium loti MAFF303099]. 3,8E-19 88,3 385
13475554 ABC transporter, ATP-binding component [Mesorhizobium loti MAFF303099]. 1E-18 87 379
13474113 hypothetical protein mll4923 [Mesorhizobium loti MAFF303099]. 1,4E-18 86,4 376
13475457 ABC transporter, polyamine transport protein, ATP-binding protein [Mesorhizobium loti MAFF303099].
4E-18 85,1 370
13475386 ABC transporter sugar ATP-binding protein [Mesorhizobium loti MAFF303099]. 2,6E-17 82,3
357
```

Outras consultas podem ser encontradas no trabalho de Otto e colegas [9].

6 Trabalhos Futuros

Nesta seção abordaremos as nossas sugestões para trabalhos futuros. Podemos citar como um primeiro trabalho futuro imediato a utilização da implementação das Estruturas Organizacionais descritas neste trabalho para todo o conjunto de dados de resultados das comparações do projeto GCP [1], cerca de 300 GB de dados comprimidos. Este trabalho futuro requer um estudo sobre a distribuição de todos os dados do conjunto de dados de resultados em outras partições, com a finalidade de podemos executar as consultas e o preenchimento das estruturas de dados para a construção das estruturas organizacionais, de uma forma mais eficiente possível. Além disso, podemos também estudar outras formas de otimização para acesso a esse volume de dados, como a distribuição do próprio banco de dados do ProteinWorlDB em várias máquinas, tendo assim um ambiente de banco de dados distribuído.

Uma outra proposta de trabalho futuro consiste na alteração da implementação atual das estruturas organizacionais, com o objetivo de transformá-la em um módulo do SGBD PostgreSQL. Assim, poderíamos distribuir a implementação do módulo tanto para o banco de dados do ProteinWorlDB quanto para outros bancos de dados de resultados de comparações de sequências, que desejem também obter as regiões ortólogas e específicas entre pelo menos dois genomas comparados.

Uma extensão ao segundo trabalho futuro proposto, está o estudo e implementação de outras metodologias para a determinação das estruturas organizacionais entre dois genomas. Nesse caso, poderíamos utilizar o algoritmo de comparação genômica RSD [23], do inglês "the Reciprocal Smallest Distance algorithm". Neste caso, teríamos que alterar tanto a metodologia para definir as regiões ortólogas, quanto a sua implementação. Neste caso, teríamos que incluir a utilização de outras ferramentas, tais como *ClustalW* (comparação de uma sequência contra um conjunto de sequências, *PAML* (obter a estimativa máxima de probabilidade de substituições de aminoácidos entre sequências de proteínas), entre outros programas necessários pelo algoritmo RSD [23].

Uma outra proposta de trabalho futuro consiste em uma remodelagem do esquema do banco de dados do ProteinWorldDB, pensando em escalabilidade. Neste caso, poderíamos utilizar alguma infraestrutura de sistema distribuído de banco de dados utilizando o framework Hadoop. Desta forma, poderíamos utilizar o data warehouse do Hive [30], que provê ferramentas para permitir facilmente a extração e carga dos dados para o ambiente do Hadoop. O Hive disponibiliza uma linguagem de consulta semelhante ao SQL, denominada de QL, que possibilita aos usuários, familiarizados com o SQL, fazerem consultas aos dados em arquivos planos. Neste caso, teríamos que instanciar o esquema de dados do ProteinWorldDB para os arquivos planos de Hive. Além disso, poderíamos implementar a tarefa de obtenção das estruturas organizacionais como tarefas *MapReduce*, sendo disponibilizadas como funções definidas pelo usuário (as UDFs). A utilização desta abordagem poderia assegurar uma melhor escalabilidade, tanto em relação a necessidade de armazenamento deste crescente volume de dados de resultados comparados, quanto ao aumento natural do poder computacional necessário a análise dessa grande quantidade de dados de genomas sequenciados recentemente. Com esta abordagem poderíamos utilizar os recursos computacionais de *Cloud Computing* para atender a essas necessidades.

7 Conclusão

Neste trabalho apresentamos a reformulação da metodologia apresentada por Viana [18] e Almeida [19] para a construção de Estruturas Organizacionais entre os genomas do Projeto de Comparação de Genomas [1]. Esta reformulação foi devido tanto às necessidades de mudanças na metodologia, por causa da ferramenta utilizada na obtenção dos resultados das comparações entre as proteínas dos genomas envolvidos, como também na necessidade de mudanças na implementação, devido a leitura e carga dos dados de resultados serem feitas a partir da base de dados do ProteinWorlDB. Neste caso, optamos por fazer pequenas modificações na implementação proposta por Viana, a fim de que as estruturas de dados utilizadas nas construções das Estruturas Organizacionais fossem preenchidas a partir da leitura dos dados de resultados do banco de dados do ProteinWorldDB.

Durante este trabalho nos deparamos com algumas dificuldades relacionadas ao gerenciamento do grande volume de dados de resultados de comparações. Algumas abordagens para a utilização mais eficiente destes dados foram também descritas neste trabalho, como a utilização de estruturas de acesso de visões materializadas, particionamento horizontal dos dados de resultados em algumas tabelas, utilizando para isso o identificador de sequências utilizadas no processo de comparação de sequências, além da proposta de utilização, como trabalho futuro, de uma infraestrutura de *data warehouse* que possibilite consultas ad-hoc, e análise desses grandes volumes de dados de uma maneira mais facilmente escalável, e também que seus dados possam ser consultados através da utilização de uma linguagem de consultas semelhante a linguagem SQL. Desta forma, sugerimos a utilização da infraestrutura do Hive como um sistema distribuído para consultas e análise dos dados de resultados do GCP, também estudando previamente a possibilidade de expansão das comparações efetuadas, para que sequências de genomas recentemente sequenciados possam ser utilizadas para a análise e construção de consulta mais complexas.

Referências Bibliográficas

[1] Genome Comparison Project - GCP. Acesso em 5 de Janeiro de 2011. URL http://www.dbbm.fiocruz.br/labwim/bioinfoteam/index.pl?action=gencomp.

[2] W. R. Pearson. Searching protein sequence libraries: Comparison of the sensitivity and selectivity of the smith-waterman and fasta algorithms. *Genomics*, 11(3):635 – 650, 1991. ISSN 0888-7543. doi: DOI:10.1016/0888-7543(91)90071-L. URL http://www.sciencedirect.com/science/article/B6WG1-4DXB5C5-1M/2/303f940c8d6bfdbc01e1de9ef037ec5a.

[3] SSEARCH - An Implementation of the SW local alignment algorithm. Acesso em 12 de Janeiro de 2011. URL http://fasta.bioch.virginia.edu/fasta_www2/fasta_list2.shtml.

[4] Temple F. Smith and Michael S. Waterman. Comparison of biosequences. *Advances in Applied Mathematics*, 2(4):482 – 489, 1981. ISSN 0196-8858. doi: DOI:10.1016/0196-8858(81)90046-4. URL http://www.sciencedirect.com/science/article/B6W9D-4D7JKHP-8G/2/800d453288c90806b562897859f9fac9.

[5] Taxonomy at National Center for Biotechnology Information - NCBI. Acesso em 12 de Janeiro de 2011. URL http://www.ncbi.nlm.nih.gov/guide/taxonomy/.

[6] The Gene Ontology - GO. Acesso em 04 de Fevereiro de 2011. URL http://www.geneontology.org/GO.downloads.database.shtml.

[7] Protein Families - Pfam. Acesso em 05 de Janeiro de 2011. URL ftp://ftp.sanger.ac.uk/pub/databases/Pfam.

[8] Kyoto Encyclopedia of Genes and Genomes - KEGG. Acesso em 03 de Fevereiro de 2011. URL http://www.genome.jp/kegg/.

[9] Thomas D. Otto, Marcos Catanho, Cristian Tristao, Marcia Bezerra, Renan M. Fernandes, Guilherme S. Elias, Alexandre C. Scaglia, Bill Bovermann, Viktors Berstis, Sergio Lifschitz, Antonio B. de Miranda, and Wim Degrave. ProteinWorldDB: querying radical pairwise alignments among protein sets from complete genomes. *Bioinformatics*, 26(5):705–707, March 2010. doi: 10.1093/bioinformatics/btq011. URL http://dx.doi.org/10.1093/bioinformatics/btq011.

[10] C. Tristão, A. B. de Miranda, and S. Lifschitz. A Conceptual Data Model Involving Protein Sets from Complete Genomes: a biological point of view. Monografias em Ciência da Computação, Departamento de Informática, PUC-Rio, 2009. URL ftp://ftp.inf.puc-rio.br/pub/docs/techreports/09_27_tristao.pdf.

[11] C. Tristão and S. Lifschitz. Protein World Database: Geração do Esquema Lógico e Processo de ETL. Monografias em Ciência da Computação, Departamento de Informática, PUC-Rio, 2009. URL ftp://ftp.inf.puc-rio.br/pub/docs/techreports/09_28_tristao.pdf.

[12] PostgreSQL. Acesso em 02 de Fevereiro de 2011. URL http://www.postgresql.org/.

[13] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990. ISSN 00222836. doi: 10.1016/S0022-2836(05)80360-2.

[14] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acid Research*, 25(17):3389–3402, 1997. ISSN 0305-1048. doi: 10.1093/nar/25.17.3389.

[15] T. A. Brown. *Genomes*. Bios Scientific Publishers Ltd; 2nd Revised edition, 2002.

[16] A. Bruce, J. Alexander, L. Julian, R. Martin, R. Keith, and W. Peter. *Molecular Biology of the Cell*. Garland Science; 4 edition, New Yourk and London, 2002.

[17] Eugene V. Koonin. Orthologs, paralogs, and evolutionary genomics. *Annual Review of Genetics*, 39(1):309–338, 2005. ISSN 0066-4197. doi: 10.1146/annurev.genet.39.073003.114725. URL `http://dx.doi.org/10.1146/annurev.genet.39.073003.114725`.

[18] C. J. M. Viana. Aspectos de genômica comparativa. Dissertação de mestrado, DCT-UFMS, 2006.

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

[20] T. F. Smith and M. S. Waterman. Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, 147(1):195 – 197, 1981. ISSN 0022-2836. doi: DOI:10.1016/0022-2836(81)90087-5. URL `http://www.sciencedirect.com/science/article/B6WK7-4DN3Y5S-24/2/b00036bf942b543981e4b5b7943b3f9a`.

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

[22] Debra L Fulton, Yvonne Y Li, Matthew R Laird, Benjamin Gs Horsman, Fiona M Roche, and Fiona Sl Brinkman. Improving the specificity of high-throughput ortholog prediction. *BMC Bioinformatics*, 7(1):270, 2006. URL `http://www.ncbi.nlm.nih.gov/pubmed/16729895`.

[23] Dennis P Wall and Todd Deluca. Ortholog detection using the reciprocal smallest distance algorithm. *Methods In Molecular Biology Clifton Nj*, 396(1):95–110, 2007. URL `http://www.ncbi.nlm.nih.gov/pubmed/18025688`.

[24] Nalvo F. Almeida, Shuangchun Yan, Magdalen Lindeberg, David J. Studholme, David J. Schneider, Bradford Condon, Haijie Liu, Carlos J. Viana, Andrew Warren, Clive Evans, Eric Kemen, Dan MacLean, Aurelie Angot, Gregory B. Martin, Jonathan D. Jones, Alan Collmer, Joao C. Setubal, and Boris A. Vinatzer. A Draft Genome Sequence of Pseudomonas syringae pv. tomato T1 Reveals a Type III Effector Repertoire Significantly Divergent from That of Pseudomonas syringae pv. tomato DC3000. *Molecular Plant-Microbe Interactions*, 22(1):52–62, 2009.

doi: 10.1094/MPMI-22-1-0052. URL http://apsjournals.apsnet.org/doi/abs/10.1094/MPMI-22-1-0052.

[25] Joao C. Setubal, Patricia dos Santos, Barry S. Goldman, Helga Ertesvag, Guadelupe Espin, Luis M. Rubio, Svein Valla, Nalvo F. Almeida, Divya Balasubramanian, Lindsey Cromes, Leonardo Curatti, Zijin Du, Eric Godsy, Brad Goodner, Kaitlyn Hellner-Burris, Jose A. Hernandez, Katherine Houmiel, Juan Imperial, Christina Kennedy, Timothy J. Larson, Phil Latreille, Lauren S. Ligon, Jing Lu, Mali Mark, Nancy M. Miller, Stacie Norton, Ina P. O'Carroll, Ian Paulsen, Estella C. Raulfs, Rebecca Roemer, James Rosser, Daniel Segura, Steve Slater, Shawn L. Stricklin, David J. Studholme, Jian Sun, Carlos J. Viana, Erik Wallin, Baomin Wang, Cathy Wheeler, Huijun Zhu, Dennis R. Dean, Ray Dixon, and Derek Wood. The genome sequence of Azotobacter vinelandii, an obligate aerobe specialized to support diverse anaerobic metabolic processes. *J. Bacteriol.*, pages JB.00504–09, 2009. doi: 10.1128/JB.00504-09. URL http://jb.asm.org/cgi/content/abstract/JB.00504-09v1.

[26] W. L. Ruzzo and M. Tompa. A Linear Time Algorithm for Finding All Maximal Scoring Subsequences. In *Seventh International Conference on Intelligent Systems for Molecular Biology*, Heidelberg, Germany, August 1999.

[27] Carlos E. R. Alves, Edson Norberto Cáceres, and Siang Wun Song. A BSP/CGM Algorithm for Finding All Maximal Contiguous Subsequences of a Sequence of Numbers. In Nagel et al. [31], pages 831–840. ISBN 3-540-37783-2.

[28] Ramez Elmasri and Shamkant Navathe. *Fundamentals of Database Systems*. Addison-Wesley Publishing Company, USA, 6th edition, 2010. ISBN 0136086209, 9780136086208.

[29] PostgreSQL/Materialized Views. Acesso em 22 de Fevereiro de 2011. URL http://tech.jonathangardner.net/wiki/PostgreSQL/Materialized_Views.

[30] Welcome to Hive! Acesso em 02 de Março de 2011. URL http://hive.apache.org/.

[31] Wolfgang E. Nagel, Wolfgang V. Walter, and Wolfgang Lehner, editors. *Euro-Par 2006, Parallel Processing, 12th International Euro-Par Conference, Dresden, Germany, August 28 - September 1, 2006, Proceedings*, volume 4128 of *Lecture Notes in Computer Science*, 2006. Springer. ISBN 3-540-37783-2.