Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Visualização de Dados Genômicos do Fungo *Paracoccidioides brasiliensis*

Marcos Francisco Ribeiro Ferreira

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Profª. Drª. Maria Emília Machado Telles Walter

Brasília
2006

Universidade de Brasília – UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado em Informática

CIP – Catalogação Internacional na Publicação

Marcos Francisco Ribeiro Ferreira .

Visualização de Dados Genômicos do Fungo *Paracoccidioides brasiliensis*/ Marcos Francisco Ribeiro Ferreira . Brasília : UnB, 2006.
101 p. : il. ; 29,5 cm.

Tese (Mestre) – Universidade de Brasília, Brasília, 2006.

1. Visualização de dados genômicos, 2. organização de genes, 3. vias metabólicas, 4. Bioinformática.

CDU 004

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro – Asa Norte
CEP 70910–900
Brasília – DF – Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Visualização de Dados Genômicos do Fungo *Paracoccidioides brasiliensis*

Marcos Francisco Ribeiro Ferreira

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Profª. Drª. Maria Emília Machado Telles Walter (Orientadora)
UnB

Prof. Dr. Nalvo Franco de Almeida Junior
UFMS

Prof. Dr. Georgios Pappas Jr.
UCB

Prof. Dr. Li Weigang
Coordenador do Mestrado em Informática

Brasília, 17 de março de 2006

Dedicatória

Dedico este trabalho à todos que nele acreditaram e o apoiaram.

Agradecimentos

Agradecimentos super especiais para minha orientadora Profª. Drª. Maria Emília, pois seus ensinamentos, orientação, paciência, compromisso e trabalho tornaram possível a realização deste sonho.

À toda minha família, em especial aos meus pais e meu irmão Márcio Júnio, que também passou por esta caminhada e sabe muito bem como ela é árdua mas ao final recompensadora.

À Cristiane Campos, minha esposa, pela sua compreensão e paciência.

À todos os meus amigos e colegas, especialmente para (em ordem alfabética): Daniela Pereira, Gleydson Fernandes, Marlene Marques, Mauro Emílio, Mônica Braga e Renata de Paiva. Obrigado pelo apoio e incentivo recebidos.

À todos servidores, mestres e doutores da UnB, em especial àqueles professores com quem tive o privilégio de ser aluno.

Ao Prof. Dr. Murilo S. Camargo pelo apoio recebido.

Por fim agradeço à Deus, que nunca me deixou sozinho nesta árdua caminhada, me dando força, paciência, esperança e motivação para vencer todos os GRANDES desafios encontrados.

"Eu Sou o Caminho a Verdade e a Vida."
Cristo Jesus

Resumo

Os projetos de seqüenciamento genômico desenvolvidos em todo o mundo geram uma enorme e crescente quantidade de informações. Estas informações estão normalmente descritas nos formatos texto, HTML (Linguagem de Marcação de Hipertexto) ou XML (Linguagem de Marcação Extensível) e seu volume total pode chegar a centenas ou milhares de páginas. Assim, o armazenamento e análise destas informações requerem o uso de ferramentas e métodos computacionais que facilitem e apóiem as pesquisas biológicas.

Além das informações de ESTs (Etiquetas de Seqüências Expressas) ou DNAs genômicos geradas pelos projetos de seqüenciamento, outros dados como comparações genômicas e genes diferencialmente expressos, obtidos por comparações usando BLAST (Ferramenta para Busca de Alinhamentos) e experimentos de macroarranjo de cDNA também são fornecidos em saídas textuais e exigem uma análise biológica. Estes conjuntos de dados são melhor analisados pelos biólogos quando representados graficamente ou através de tabelas. Neste contexto, este trabalho propõe dois métodos para visualização de dados gerados pelos projetos de seqüenciamento genoma realizados pelo Laboratório de Biologia Molecular da Universidade de Brasília - UnB.

Comparações obtidas pelo BLAST entre as ESTs do fungo *Paracoccidioides brasiliensis* com os DNAs genômicos de dois outros fungos próximos filogeneticamente, o *Aspergillus fumigatus* e o *Aspergillus nidulans*, utilizando a ferramenta para visualização de dados genômicos GBrowse, permitirão inferir a organização dos genes do *P. brasiliensis* dentro dos seus cromossomos. Particularmente, a partir do modelo *A. fumigatus*, sintenias entre os genes do *P. brasiliensis* poderão ser identificadas. A partir destas inferências, experimentos biológicos poderão ser elaborados para confirmação da organização dos genes do *P. brasiliensis*.

Por outro lado, visando contribuir para a análise do conjunto de genes diferenciais indicados por experimentos de macroarranjo de cDNA, implementamos a ferramenta *PathwayView* para localização destes genes nos mapas de vias metabólicas do KEGG (Enciclopédia de Genes e Genomas da Universidade de Kioto). A *PathwayView* contribuirá para processar um grande volume de dados, facilitando a localização das vias metabólicas e das enzimas diferenciais dentro destas vias. No caso específico do *P. brasiliensis*, esta ferramenta permitirá aprofundar, por

exemplo, o conhecimento de como ocorre a adaptação biológica deste patógeno nos seres humanos, o que poderá indicar direções novas para o desenvolvimento de drogas ou fungicidas.

Palavras-chave: Visualização de dados genômicos, organização de genes, vias metabólicas, Bioinformática.

Abstract

Genome sequencing projects developed worldwide produce an enormous and growing volume of information. These informations are usually described in text, HTML (Hypertext Markup Language) or XML (Extensible Markup Language) format, and they can be stored on hundreds or even thousands of pages. The storage and analysis of these information require computational tools and methods to facilitate and support the biological research.

Besides the EST (Expressed Sequence Tags) information or genomic DNAs generated from the sequencing projects, other data such as genomic comparisons or differential gene expression, obtained from BLAST (Basic Alignment Search Tool) and cDNA macroarray experiments are available on text format which requires biological analysis. The work of the biologists can be supported if the available data is presented using graphics or tables. So, this work proposes two methods for visualizing data generated by the sequencing genome projects developed on the molecular biology of the University of Brasilia.

Comparisons obtained by BLAST among the *Paracoccidioides brasiliensis* ESTs and the genomics DNAs of two others phylogenetic related fungi, *Aspergillus fumigatus* and *Aspergillus nidulans*, using a genomic visualization tool - GBrowse, allow to infer the organization of the *P. brasiliensis* genes inside its chromosomes. Particularly, from the model *A. fumigatus*, synteny among the *P. brasiliensis* genes could be identified. The inferences obtained by these comparisons can greatly help to develop biological experiments to confirm the organization of the *P. brasiliensis* genes.

Besides, to contribute for the analysis of the set of differentially expressed genes indicated by cDNA macroarray experiments, a tool named *PathwayView* was built, to locate the differential genes into the KEGG metabolic pathways maps. *PathwayView* contribute to process a great volume of data, making easier to find the pathways and the enzymes inside these pathways. In the particular case of the *P. brasiliensis* fungus, for example, this tool could be used to increase the knowledge of how biological adaptation of this pathogen occurs in humans, which in turn, could indicate new directions for the development of new drugs or fungicides.

Keywords: Genomic data visualization, genes organization, metabolic pathways, Bioinformatics.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Após a proposta da estrutura do DNA apresentada por James Watson e Francis Crick [60], um grande esforço foi feito pela comunidade científica com o objetivo de compreender a estrutura e o funcionamento da genética dos seres vivos. Recentes avanços nas técnicas da Biologia Molecular e na Bioinformática permitiram acelerar o processo de descoberta e descrição da estrutura e funcionalidade dos genes. Em particular, o seqüenciamento em massa e o desenvolvimento de ferramentas computacionais para análise comparativa entre seqüências, disponibilizaram um grande volume de informações, propiciando um avanço rápido nas pesquisas de Biologia Molecular, além do desenvolvimento de novos algorítmos em Ciência da Computação.

Este crescente volume de informações está normalmente disponibilizado em formatos baseados em texto, dificultando a análise por parte dos biólogos. Para auxiliar nesta análise, facilitando a interpretação destas informações, faz-se necessário disponibilizar a estes profissionais, ferramentas que possibilitem a visualização destas informações em formatos não puramente textuais, como tabelas e gráficos interativos.

Atualmente existem várias ferramentas com este objetivo. Estas ferramentas devem ser exploradas para beneficiar de forma rápida os Projetos Genomas existentes, pois a maioria delas incorpora um grande conjunto de funções implementadas de forma bastante eficiente.

Por outro lado, ferramentas que auxiliem a visualização de informações ainda precisam ser desenvolvidas quando se deseja trabalhar com vias metabólicas armazenadas no formato definido pelo *Kyoto Encyclopedia of Genes and Genomes* - KEGG [32, 33]. Neste caso, o número de ferramentas disponíveis é bastante escasso e muitos processos acabam sendo realizados pelos biólogos de forma manual. Para se construir ferramentas que utilizem recursos como as vias metabólicas KEGG, faz-se necessária a utilização de interfaces de programas definidas também

pelo KEGG. Através destas interfaces, ferramentas cliente podem ter acesso às informações contidas nos bancos de dados remotos, realizando consultas e processamentos nestes dados.

Portanto é muito importante conhecermos as ferramentas de visualização existentes em Bioinformática, afim de utilizarmos seus recursos em benefício dos trabalhos em Biologia Molecular.

1.1 Contextualização

Por todo o mundo existem vários centros públicos especializados no seqüenciamento de genomas, entre os quais podemos citar o TIGR (The Institute for Genomic Research) [55], o Sanger Centre [48], o DOE Joint Genome Institute (JGI) [20] e o Broad Institute do MIT [13] entre outros.

Estes centros foram sendo criados a partir do Projeto Genoma Humano, responsável pelo seqüenciamento do DNA humano. Duas principais frentes de pesquisas se empenharam paralelamente para a execução deste projeto: o Consórcio Internacional de Seqüenciamento do Genoma Humano - IHGSC, formado por 20 centros em 6 países, e a empresa privada Celera Genomics.

Ambos os projetos divulgaram em fevereiro de 2001 as primeiras versões de todo o genoma humano [17, 57]. O resultado final divulgado pelo IHGSC, em maio de 2004, continha informações sobre 2,85 bilhões de nucleotídeos identificados [18]. Informações como estas ficam armazenadas em enormes bancos públicos de genomas como o GenBank [39], o Banco de DNA do Japão - DDBJ [19] e o Laboratório de Biologia Molecular Europeu - EMBL [23].

No Brasil, a FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo) foi a primeira instituição a apoiar o desenvolvimento de projetos de seqüenciamento de genomas. Formado pelo Laboratório de Bioinformática (LBI) da Unicamp e por um conjunto de laboratórios de Biologia Molecular do estado de São Paulo, foi criado o instituto virtual responsável pelo seqüenciamento e análise de nucleotídeos, denominado ONSA (Organization for Nucleotide Sequencing and Analysis) [40]. O primeiro resultado importante desse instituto, bem como o seu reconhecimento internacional, ocorreu com a publicação do genoma do fitopatógeno *Xylella fastidiosa* [51], agente etiológico (causador) da *Citrus Variegated Chlorosis* (CVC), mais conhecida com "praga do amarelinho". Esta doença destrói lavouras de laranja, ocasionando prejuízos econômicos de grandes proporções.

Após o projeto da *X. fastidiosa*, a FAPESP investiu em outros projetos [40],

como o do mapeamento do genoma da cana-de-açúcar, do câncer humano (em colaboração com o Instituto Ludwig para Pesquisa do Câncer), do café e também de vários organismos e pragas como a *Xylella fastidiosa de videira*, a *Xanthomonas campestris*, a *Xanthomonas axonopodis* e a *Leifsonia syli*, além de subsidiar outros projetos como o do mapeamento do genoma funcional do *Shistosoma mansoni*.

O governo federal apoiou uma rede de seqüenciamento em âmbito nacional, chamado Projeto Genoma Brasileiro [45], que seqüenciou a bactéria *Chromobacterium violaceum*, além de ter induzido diversos projetos regionais, como a Rede Genoma do Estado de Minas Gerais, para o seqüenciamento do *Schistosoma mansoni*, a Rede Genoma Nordeste, para o seqüenciamento do *Leishmania chagasi*, a Rede Genoma do Estado da Bahia e São Paulo, para o seqüenciar a *Crinipellis perniciosa* e também a Rede Genoma Centro-Oeste, responsável por seqüenciar o fungo *Paracoccidioides brasiliensis*.

O Projeto BIOFOCO

Na região Centro-Oeste, foi criada em 2002 uma rede de pesquisa e desenvolvimento em Bioinformática, denominada Projeto BIOFOCO [10]. A primeira fase deste projeto teve como objetivo geral, integrar as instituições de pesquisa e ensino desta região e oferecer apoio aos grupos de pesquisa genômica e proteômica, através de troca de conhecimentos, ferramentas, sistemas e capacitação de técnicos e pesquisadores. Seus objetivos específicos foram:

- Selecionar, desenvolver e integrar ferramentas e sistemas para anotação automática de genoma e proteoma;
- Ampliar a disponibilização de bases de dados próprias e públicas e serviços de Bioinformática à rede de parceiros e à comunidade científica;
- Fortalecer a estruturação de atividades de ensino e pesquisa em Bioinformática;
- Fortalecer a capacitação de pesquisadores e técnicos em Bioinformática, para desenvolvimento de ferramentas de análise, integração de sistemas e suporte computacional às pesquisas genômica e proteômica aplicadas.

Atualmente, este projeto está na segunda fase, cujo objetivo principal é consolidar a rede de pesquisa em Bioinformática, construindo um serviço de *Grid computacional*[1] para anotação genômica, que permitirá uma melhor distribuição e

[1]Conjunto de recursos como supercomputadores, estações de trabalho e bancos de dados, geograficamente distribuídos e compartilhados entre usuários.

alocação de recursos computacionais e também facilitará a integração de aplicações, pesquisas e atividades de Bioinformática das instituições envolvidas no projeto.

Esta segunda fase do projeto continua sob a mesma coordenação e inclui as participações da UFG, UFMS, UFAL, UFRGS e LGE/UNICAMP.

O Projeto Genoma *Paracoccidioides brasiliensis*

Apresentaremos com mais detalhes o Projeto Genoma do *Paracoccidioides brasiliensis*, pois utilizaremos dados deste projeto para realizarmos os experimentos do presente trabalho.

O *Paracoccidioides brasiliensis* é o agente etiológico da *paracoccidioidomicose* - PCM, uma doença crônica e progressiva que ataca, entre outros órgãos humanos, pulmões, mucosa da boca e nariz, afetando predominantemente comunidades nas Américas do Sul e Central.

O *Paracoccidioides brasiliensis* é um fungo dimórfico (possui duas formas): à temperatura ambiente, aproximadamente $\sim 26^o$ C, ele é encontrado na forma de micélio (M) e em temperaturas próximas a $\sim 36^o$ C é encontrado na forma de levedura (L) (Figura 1.1).

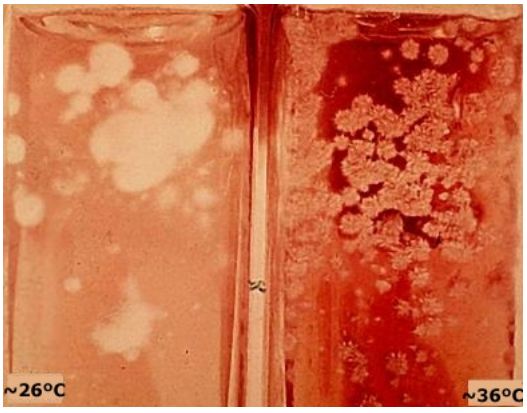


Figura 1.1: O fungo *Paracoccidioides brasiliensis* na forma de micélio ($\sim 26^o$) e levedura ($\sim 36^o$) [58].

A PCM é contraída principalmente pela inalação de fragmentos de micélio e quando infecta o hospedeiro o *Paracoccidioides brasiliensis* passa para a forma de levedura, provavelmente ocasionado pela mudança de temperatura.

O dimorfismo apresenta-se como uma característica comum a fungos patogênicos humanos e é um processo ainda pouco compreendido. Assim, o estudo de genes diferenciais nas formas de micélio e de levedura poderá contribuir para identificar genes que constituem potenciais alvos para drogas.

A equipe do Projeto Genoma Funcional e Diferencial do fungo *Paracoccidioi-*

des brasiliensis - Projeto Genoma Pb [44], formada por pesquisadores e alunos da Universidade de Brasília - UnB, Universidade Federal de Goiás - UFG, Universidade Federal do Mato Grosso - UFMT, Universidade Federal do Mato Grosso do Sul - UFMS e várias outras instituições vêm pesquisando os diferentes aspectos da biologia molecular do *Paracoccidioides brasiliensis*.

Como resultados do Projeto Genoma Pb, foram relatadas 19.718 seqüências genômicas selecionadas segundo critérios de tamanho e qualidade de seqüenciamento, tais que 9.777 são seqüências de levedura e 9.941 são de micélio. A partir destas seqüências foram gerados 6.022 grupos, contendo 2.655 *contigs* e 3.367 *singlets*. Os *contigs* são formados por seqüências com regiões sobrepostas e uma única seqüência (seqüência consenso), formada a partir das sobreposições das seqüências, representa todas as seqüências deste grupo. Os *singlets* são constituídos por seqüências que não apresentaram similaridade mínima com nenhuma outra do conjunto inicial de seqüências.

Outros Organismos Relacionados no Trabalho

Outros dois fungos serão utilizados neste trabalho: o *Aspergillus fumigatus* e *Aspergillus nidulans*.

O *Aspergillus fumigatus* (Figura 1.2) é um fungo altamente oportunista e as infecções que ele causa nos seres humanos é o segundo maior problema de saúde nos Estados Unidos [12]. Devido à sua importância, o *Aspergillus fumigatus*, tem sido seqüenciado por institutos como TIGR [55] e o Sanger Centre [48].

Figura 1.2: O fungo *Aspergillus fumigatus* [7].

O tamanho do genoma do *Aspergillus fimigatus* é de aproximadamente 30 milhões de bases, organizados em 8 cromossomos [8].

O *Aspergillus nidulans* (Figura 1.3), também conhecido como *Emericella nidulans*, é um fungo bastante utilizado como organismo modelo para o entendimento

de várias questões biológicas [14] e está relacionado a uma grande quantidade de outras espécies de *Aspergillus*, como o *A. oryzae*, *A. favus* e *A. fumigatus*.

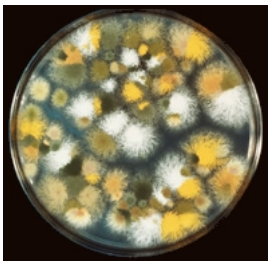


Figura 1.3: O fungo *Aspergillus nidulans* [14].

O tamanho do genoma do *Aspergillus nidulans* é de aproximadamente 31 milhões de bases, contendo 8 cromossomos e cerca de 11 mil a 12 mil genes [14].

1.2 Motivação

Os projetos de seqüenciamento geram uma enorme e crescente quantidade de dados, auxiliados por ferramentas de comparação de seqüências como FASTA [42, 41] e BLAST [3, 4], usadas rotineiramente para procurar, nos bancos de dados internacionais de seqüências biológicas, seqüências similares àquelas geradas no âmbito do projeto de seqüenciamento.

As ferramentas de comparação de seqüências produzem dados normalmente nos formatos texto, HTML[2] ou XML, onde o volume total pode chegar a centenas ou milhares de páginas. Outro problema relevante nas análises biológicas baseadas nestes dados é que as informações encontram-se dispersas, dificultando a interpretação por parte dos biólogos.

Os bancos de dados internacionais de anotação de seqüências, como o GenBank, o DDBJ e o EMBL, sofrem um vertiginoso aumento na quantidade de seqüências depositadas dos programas de comparação de seqüências (Figura 1.4). O NCBI anunciou em agosto de 2005, que juntos, estes bancos de seqüências de DNA excederam 100 gigabases (100 bilhões de pares de bases) [39]. Com esta grande quantidade de seqüências, a tarefa de analisar os dados de saída torna-se difícil, já que os mesmos tendem também a crescer, devido a disponibilidade de mais seqüências similares serem encontradas nestes bancos de dados.

Assim, pesquisas têm mostrado que a visualização das informações através de gráficos interativos ou apresentações visuais (não textuais) são métodos mais efetivos para auxiliar na interpretação de grandes quantidades de dados científicos

[2]Linguagem de Marcação de Hipertexto, utilizada para publicação de páginas na *Internet*.

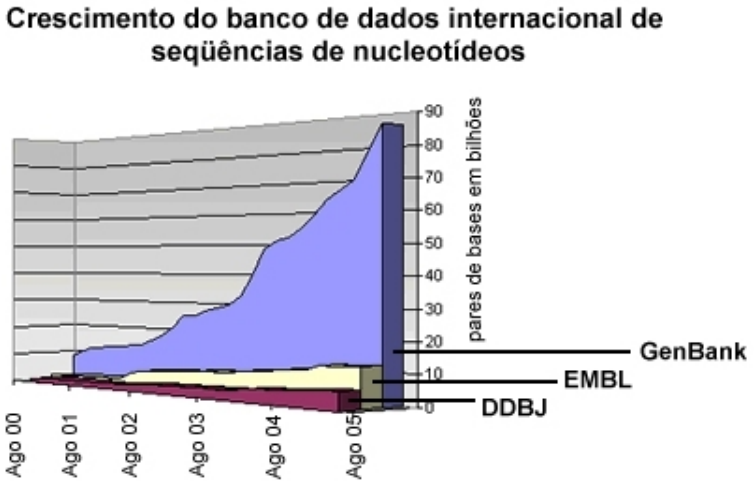


Figura 1.4: Estatística de crescimento dos bancos internacionais de seqüências de nucleotídeos [39].

[15, 35]. Faz-se portanto, necessário investir na utilização e criação de ferramentas de visualização para apoiar e agilizar a análise de dados sem perda de informação.

1.3 Objetivos

O objetivo geral deste trabalho é disponibilizar visualização de dados dos projetos genoma realizados pelo Laboratório de Biologia Molecular da Universidade de Brasília - UnB, especialmente do fungo *Paracoccidioides brasiliensis*. Particularmente neste trabalho visamos:

1. Desenvolver um forma de visualização de dados genômicos que visa exibir comparações entre seqüências de diferentes organismos, realizadas através do programa de comparação de seqüências BLAST.

2. Desenvolver uma ferramenta computacional para localização de enzimas em mapas de vias metabólicas e para a visualização destes mapas.

3. Usar a visualização de dados genômicos comparados por BLAST para os fungos *Paracoccidioides brasiliensis*, *Aspergillus fumigatus* e *Aspergillus nidulans*.

4. Usar a ferramenta de visualização em mapas de vias metabólicas do KEGG [33] para os genes diferencialmente expressos de *Paracoccidioides brasiliensis*, identificados por experimentos de microarranjo de cDNA.

1.4 Organização do Trabalho

O Capítulo 2 apresenta as ferramentas de anotação genômica BLAST e KEGG. O Capítulo 3 descreve uma ferramenta para visualização de dados genômicos - GBrowse e mostra sua utilização com dados obtidos de um projeto genoma. O Capítulo 4 apresenta um método para visualização de dados genômicos utilizando o GBrowse, além de descrever e discutir experimentos deste método para os fungos *Paracoccidioides brasiliensis*, *Aspergillus fumigatus* e *Aspergillus nidulans*. O Capítulo 5 apresenta uma ferramenta para localização e visualização de genes diferencialmente expressos em mapas de vias metabólicas do KEGG, além de mostrar e discutir experimentos com *Paracoccidioides brasiliensis* nesta ferramenta. Finalmente o Capítulo 6 apresenta as conclusões e sugere trabalhos futuros.

Capítulo 2

Ferramentas de Anotação

Nesta capítulo, descreveremos as ferramentas de anotação BLAST e KEGG, utilizadas neste trabalho. O BLAST é utilizado para realizar as comparações entre seqüências biológicas. Os resultados destas comparações integram o método que será proposto no Capítulo 4. Já o KEGG é um conjunto de recursos, como APIs[1], bancos de dados e estruturas em XML, que são utilizados na construção da ferramenta de visualização de enzimas em mapas de vias metabólicas, que será apresentada no Capítulo 5.

A Seção 2.1 apresenta a ferramenta BLAST, abordando a comparação de seqüências, o funcionamento do algorítmo BLAST e os principais programas BLAST. A Seção 2.2 apresenta a ferramenta KEGG, fazendo uma pequena introdução sobre XML, a KGML, a API KEGG, SOAP e WSDL.

2.1 BLAST

O BLAST [3, 4] - *Basic Alignment Search Tool* (Ferramenta para Busca de Alinhamento) é um conjunto de programas computacionais para comparação de seqüências. Basicamente a comparação é feita entre uma seqüência (seqüência de busca) e um banco de dados de seqüências (seqüências alvo) previamente formatado.

2.1.1 Comparação de Seqüências

A comparação de seqüências é uma das operações mais elementares em Bioinformática e é amplamente utilizada em pesquisas e projetos de seqüenciamento, servindo como base para a solução de uma grande variedade de problemas [49].

[1] Application Program Interface: camada de software que disponibiliza um conjunto de rotinas padronizadas para ser utilizada por outros programas.

Consideremos as seqüências GACGGATTAG e GATCGGAATA. Por meio de uma inspeção visual podemos notar que essas seqüências são bem semelhantes entre si, pois coincidem em vários caracteres. Ao efetuar essa análise, no entanto, não nos preocupamos em quantificar essa semelhança. Para explicitar e quantificar semelhanças e diferenças existentes entre as duas seqüências é necessário um mecanismo capaz de computar uma medida que denote quanto estas seqüências são parecidas (similares). Para calcular esse grau de similaridade, os pesquisadores usam a noção de alinhamento de seqüências.

O alinhamento de seqüências está associado à idéia de dispor as seqüências a serem analisadas uma sobre a outra, de forma a explicitar a correspondência dos caracteres nas seqüências. Para o alinhamento, são utilizadas mutações pontuais nos genes tais como substituições, remoções ou inserções de bases. Uma distância entre as duas seqüências pode ser então computada associando custos a essas mutações. Como se deseja encontrar a menor distância possível, deve-se alinhar as seqüências de modo que o número dessas mutações seja o menor possível. No alinhamento, posicionamos os caracteres das duas seqüências de modo que um caractere em uma seqüência esteja alinhado com um caractere ou espaço na outra seqüência. Esses espaços são utilizados para permitir uma melhor correspondência entre caracteres nas duas seqüências e garantem que, no final, as seqüências alinhadas tenham o mesmo comprimento. Na Figura 2.1 exibimos um possível alinhamento entre as seqüências citadas anteriormente. Por convenção, nenhum espaço em uma seqüência é alinhado com um espaço na outra seqüência [49].

```
G  A  -  C  G  G  A  T  T  A  G
G  A  T  C  G  G  A  A  T  A  -
```

Figura 2.1: Exemplo de alinhamento entre duas seqüências. O espaço *(Gap)* é representado pelo caractere "-".

A noção de similaridade de seqüências está associada à idéia de caracterizar, através de uma pontuação, um particular alinhamento de seqüências. Usualmente consideramos a correspondência (ou não) de caracteres no alinhamento para computar essa pontuação. Seja L um alinhamento qualquer. Definimos uma pontuação para uma coluna de L atribuindo a ela um valor pré-definido, de acordo com a correspondência ou não de caracteres nessa posição do alinhamento. Para uma coluna contendo caracteres idênticos poderíamos, por exemplo, atribuir o valor +1, a uma coluna contendo caracteres diferentes atribuir o valor -1 e a uma coluna contendo um espaço poderíamos atribuir o valor -2. Somando-se a pontuação individual de cada coluna, obtém-se a pontuação final do alinhamento.

A Figura 2.2 mostra o alinhamento da figura anterior, indicando a pontuação de cada coluna como definimos anteriormente.

```
G   A   -   C   G   G   A   T   T   A   G
G   A   T   C   G   G   A   A   T   A   -

+1  +1 -2  +1  +1  +1  +1  -1  +1  +1  -2  = +3
```

Figura 2.2: Alinhamento utilizando a seguinte pontuação: +1 para os caracteres que coincidem, -1 para os que não coincidem e -2 para um alinhamento entre caractere e espaço. A pontuação final para este alinhamento é +3.

O melhor alinhamento é aquele que, dentre todos os alinhamentos possíveis, possui a pontuação mais alta. A similaridade entre duas seqüências pode ser definida então como sendo a pontuação do melhor alinhamento entre essas seqüências. Um modo de determinar a similaridade entre duas seqüências seria gerar todos os alinhamentos possíveis e, daí, escolher o melhor deles. Entretanto, o número de alinhamentos gerados é exponencial tornando essa abordagem inviável na prática.

2.1.2 Algoritmo BLAST

Ao realizar a comparação de seqüências, o BLAST retorna uma lista de **HSPs** (High-scoring segment pairs), ou pares de segmentos de alto escore, contendo as maiores similaridades entre segmentos das seqüências busca e alvo.

Um segmento na terminologia BLAST é um fragmento *(subseqüência)* de uma seqüência. Dadas duas seqüências, um *par de segmentos* é formado por fragmentos de mesmo comprimento pertencentes a cada uma das seqüências.

Como os fragmentos nestes pares de seqüências possuem o mesmo tamanho, um alinhamento sem *gaps* é formado entre elas. O alinhamento pode ser pontuado usando uma matriz de pontuação, como a PAM120 (Figura 2.3).

```
 K  A  L  M  R
 V  A  K  N  S
-4  3 -4 -3 -1  =  -9
```

Figura 2.3: Pares de segmentos e sua pontuação utilizando a matriz de pontuação PAM120.

O par de segmento máximo (MSP) entre duas seqüências é um par de segmento de maior pontuação. Esta pontuação é uma medida de similaridade entre seqüências e pode ser precisamente computada através de programação dinâmica. Porém, o BLAST consegue estimar este número de forma mais rápida que qualquer método de programação dinâmica.

Para calcular os pares de segmentos de maior pontuação, BLAST procede da seguinte maneira: encontra "sementes", que são pares de segmentos muito curtos. As sementes são estendidas em ambas as direções, sem a inclusão de gaps, até que a pontuação máxima possível seja encontrada. Quando uma pontuação atinge um limite inferior, o programa finaliza a extensão daquela semente.

BLAST pode ser resumido em 3 procedimentos algoritmicos descritos a seguir:

1. Compilação da lista de maior pontuação de seqüências (ou *palavras*);
2. Busca por alinhamentos entre caracteres - cada alinhamento gera uma *semente*;
3. Estende as sementes geradas.

Os passos que o algoritmo executa dependem de qual tipo de comparação de seqüência é realizada: DNA ou proteína.

Para seqüências de proteína, a lista de maior pontuação consiste de todas as palavras com w caracteres *(w-mers)* que pontuaram pelo menos T com certo *(w-mer)* da consulta, usando uma matriz de pontuação PAM. Os parâmetros w e T podem ser ajustados na execução do programa.

Existem duas abordagens que são utilizadas pelo BLAST para busca da lista de pontuação máxima no banco de dados: uma delas é arranjar esta lista de palavras em uma tabela *hash* e realizar a procura através do índice desta tabela. A segunda é fazer uso de um autômato finito determinístico. Este método inicia um estado fixo e para cada caractere no banco de dados é realizada uma transição de estado. Dependendo do estado e da transição, uma palavra da lista é reconhecida.

Para seqüências de DNA, a lista de maior pontuação contém somente as consultas *w-mers*. A estratégia para percorrer o banco de dados é bastante diferente. Como o alfabeto é pequeno, ou seja, possui tamanho 4, o banco de dados é primeiro comprimido de tal forma que cada nucleotídeo seja representado por 2 bits, ou 4 nucleotídeos por 1 byte. Assim, a procura pode ser realizada de forma mais rápida, comparando um byte por vez.

2.1.3 Programas BLAST

O BLAST possui vários programas para comparação de seqüências e programas auxiliares como o *formatdb*, que realiza a formatação de um banco de dados BLAST para comparação: dado um arquivo, normalmente no formato FASTA, contendo seqüências de nucleotídeos ou proteínas, o *formatdb* gera um banco de dados BLAST correspondente.

A escolha do programa de comparação apropriado é influenciado por fatores como os tipos da seqüência de busca e do banco de dados alvo. A Tabela 2.1 mostra alguns exemplos.

Seqüência de Busca	**Banco de Dados**	**Programa**	**Descrição**
Nucleotídeo	Nucleotídeo	BLASTN	Encontra similaridades entre seqüências de nucleotídeos (busca e banco de dados).
Proteína	Proteína	BLASTP	Encontra similaridade entre seqüências de proteínas (busca e banco de dados).
Nucleotídeo	Proteína	BLASTX	Traduz as seqüências de busca de nucleotídeos para proteínas e as compara com as seqüências de proteínas do banco de dados.
Nucleotídeo	Nucleotídeo	TBLASTX	Traduz as seqüências de busca e as seqüências do banco de dados de nucleotídeos para proteínas e as compara.
Proteína	Nucleotídeo	TLASTN	Compara as seqüências de busca de proteínas com as seqüências do banco de dados que são traduzidas de nucleotídeos para proteínas.

Tabela 2.1: Seleção de programas BLAST de acordo com a seqüência de busca e o tipo do banco de dados.

O experimento realizado neste trabalho utilizou o BLAST executando os programas BLASTN e TBLASTX.

2.2 KEGG

O *Kyoto Encyclopedia of Genes and Genomes* - KEGG [32] é um conjunto de programas e informações de Bioinformática, que vem sendo desenvolvido pelo Laboratório Kanehisa da Universidade de Kioto no Japão.

O objetivo do KEGG é contribuir para a solução de problemas da chamada *era pós-genômica*, como a representação computacional das células e dos organismos entre outros [33].

Atualmente o KEGG é constituído por quatro banco de dados, conforme mostra a Tabela 2.2.

Banco de dados	**Conteúdo**
Vias	informações referentes aos mapas de vias
Genes	informações sobre genes, como nome, anotação e classificação de genomas
Ligand	informações sobre combinações químicas, enzimas e reações enzimáticas
BRITE	informações sobre classificação de drogas entre outros

Tabela 2.2: Bancos de dados que compõem o KEGG [33].

Particularmente neste trabalho estamos interessados no banco de dados de vias, que contém entre outros informações sobre vias metabólicas.

Uma via metabólica é uma série de reações enzimáticas conectadas entre si e que produzem um produto genético específico [2].

Os dados funcionais relacionados a seqüências de DNAs ou proteínas estão atualmente armazenados como anotações. Estas anotações basicamente representam a função de uma única molécula, isto é, um componente individual de um sistema biológico, não contendo portanto informações de alto nível como diagramas que representem interações genéticas e moleculares. Sem diagramas como estes um sistema biológico não pode ser descrito e entendido. No KEGG cada via metabólica é representada por um diagrama chamado mapa de vias.

A Figura 2.4 mostra o mapa de vias metabólicas dos aminoácidos glicina, serina e treonina, pertencente ao sub-grupo do metabolismo dos aminoácidos, que pertence ao grupo de metabolismo.

Os mapas de vias são formados de uma série de elementos gráficos, representando as enzimas, as combinações e outros mapas (Figura 2.5). Cada enzima possui no KEGG um código chamado Enzyme Commission Number - EC, que é utilizado para sua identificação nos mapas de vias.

A seguir faremos uma breve descrição da tecnologia XML [63], pois a mesma está presente nos dois recursos fornecidos pelo KEGG que serão apresentados posteriormente: a KGML e a API *Web Service*.

2.2.1 XML

A XML - *Extensible Markup Language* [63] ou Linguagem de Marcação Extensível é uma meta-linguagem utilizada para a construção de outras linguagens de marcação especializadas.

Uma marcação *(tag)* é um símbolo adicionado ao documento XML que identifica partes deste documento e sua relação com outras partes [46]. Na Figura 2.6 temos o exemplo de um documento XML, onde as tags <email> e </email> marcam o início e fim do documento. As tags <destinatarios> e </destinatarios> envolvem a informação dos destinatários, incluindo as tags <para>, </para>, <copia> e </copia> que informam o email de cada destinatário.

O formato XML é bastante utilizado como padrão para troca de informações na Web e fora dela. A seguir são listadas algumas das principais características da XML:

- XML é uma padrão aberto;

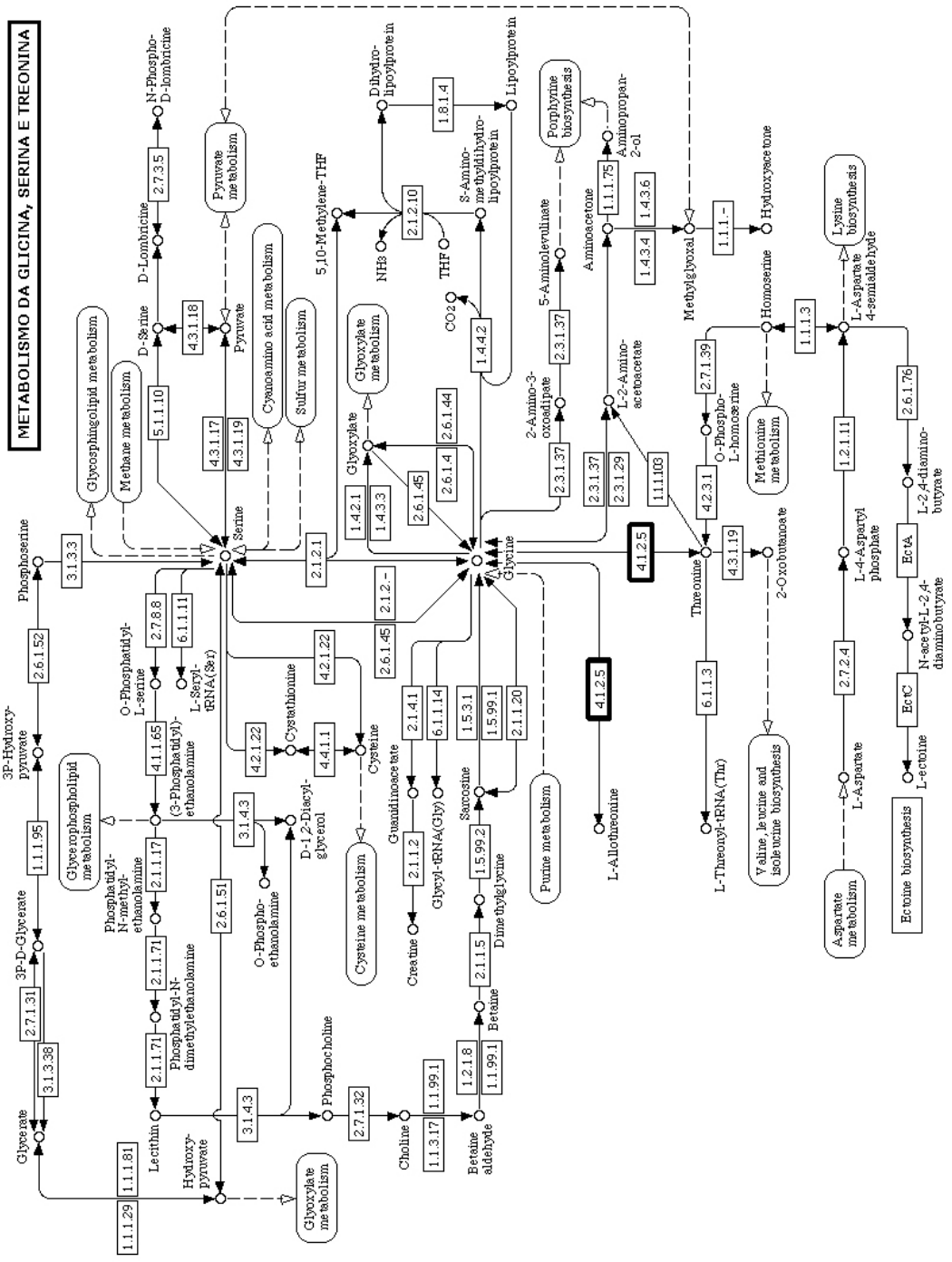


Figura 2.4: Mapa de vias metabólicas dos aminoácidos glicina, serina e treonina [33], observando-se que os números nas caixas retângulares representam o código da enzima (*Enzyme Commission Number - EC*). Por exemplo, o EC:4.1.2.5 é o código da enzima *threonine aldolase*.

- Pode armazenar e organizar praticamente qualquer tipo de informação em um documento XML;
- É baseado no conjunto de caracteres Unicode, suportando assim uma grande

Figura 2.5: Elementos gráficos utilizados na construção dos mapas de vias metabólicas KEGG [33].

```
<email>
<autor>Nome do autor</autor>
<destinatarios>
<para>pessoa@email</para>
<copia>outrapessoa@email</copia>
</destinatarios>
<mensagem>texto do email</mensagem>
</email>
```

Figura 2.6: Trecho de um documento XML.

gama de símbolos e caracteres de diversas línguas;

- Oferece mecanismos para verificação de sintaxe, regras e tipos de dados entre outros em um documento XML;
- Possui uma sintaxe e estrutura fácil de ser lido inclusive por seres humanos.

2.2.2 KGML

A KEGG Markup Language - KGML é um formato em XML utilizado pelo KEGG para descrever seus objetos gráficos, especialmente os mapas de vias.

Através das informações representadas com a KGML, podemos criar ferramentas para análises computacionais e construção automática e dinâmica de mapas de vias. O modelo de dados da KGML é mostrado no diagrama da Figura 2.7.

Na KGML, o elemento *pathway* especifica um objeto gráfico composto por nós, reações e relações, que são representados respectivamente pelos elementos *entry, reaction* e *relation*. O elemento *relation* indica conexão entre proteínas (produtos genéticos) e o elemento *reaction* indica conexão entre combinações químicas. O elemento *graphic* define informações sobre os objetos gráficos e *component* é utilizado para referência recursiva entre elementos *entry*. Os elementos *product* e *substrate* especificam respectivamente o produto e substrato da reação, já o elemento *alt* permite atribuir nomes alternativos para estes dois elementos. Por último, o elemento *subtype* permite especificar mais informações sobre a relação.

A KGML, assim como grande parte dos arquivos XML, está definida através de um Document Type Definition - DTD, ou Definição de tipo de documento,

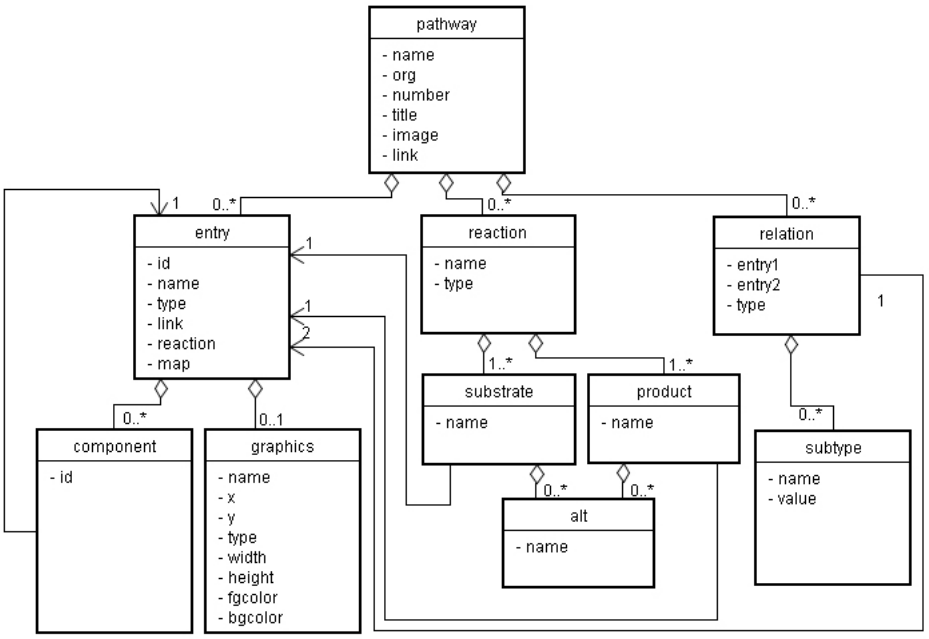


Figura 2.7: Diagrama do modelo de dados da KGML [33].

que descreve a estrutura de elementos de um arquivo XML. O DTD da KGML está disponível pela *Internet* [33].

2.2.3 API KEGG

O KEGG provê uma *Application Program Interface* - API que permite que outras aplicações interajam com o sistema, realizando operações como consultas e solicitação de mapas de vias entre outras. A API KEGG é disponibilizada na forma de *Web Service*.

Podemos caracterizar um *Web Service* como um componente de serviços contendo lógicas de negócio localizado em algum ponto na Internet, acessado através do protocolo HTTP, que utiliza a XML para troca de mensagens [16].

Atualmente, três tecnologias têm emergido como padrão para compor o núcleo do *Web Service*: SOAP [52], WSDL [62] e UDDI [56].

SOAP: O *Simple Object Access Protocol* [52] é um protocolo desenvolvido para permitir a troca de informações estruturadas em ambientes distribuídos. SOAP utiliza a tecnologia XML para definir um *framework* que permite criar mensagens que possam ser trocadas entre aplicações através do protocolo HTTP. Estas aplicações podem estar sendo executadas em diferentes sistemas operacionais, tecnologias e linguagens de programação.

Uma mensagem SOAP é um documento XML contendo:

- Um elemento chamado Envelope, que identifica o documento XML como uma mensagem SOAP;
- Um elemento chamado Body, que contém informações da chamada e resposta da mensagem;
- Um elemento opcional chamado *Header*, que contém informações específicas da aplicação;
- Um elemento opcional chamado *Fault*, que provê informações sobre erros que podem ocorrer durante o processamento da mensagem.

A Figura 2.8 mostra a estrutura de uma mensagem SOAP, contendo os quatro elementos acima.

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
...
</soap:Header>
<soap:Body>
...
<soap:Fault>
...
</soap:Fault>
</soap:Body>
</soap:Envelope>
```

Figura 2.8: Estrutura de uma mensagem SOAP.

WSDL: O *Web Service Description Language* [62] é uma linguagem baseada no formato XML que descreve as interfaces, as operações, os tipos de dados e a localização do *Web Service*.

Assim, para que um programa cliente possa acessar um *Web Service* e realizar uma troca de mensagem, é necessário primeiramente obter o arquivo WSDL do *Web Service* desejado e em seguida construir a mensagem de acordo com as informações (como nome da mensagem, parâmetros e tipo de retorno) contidas neste arquivo.

O *Web Service* KEGG possui em seu arquivo WSDL diversas funcionalidades para processamento de informações armazenadas em seus bancos de dados. Algumas destas funcionalidades estão listadas no Apêndice A.

UDDI: *Universal Description, Discovery and Integration* [56] provê um registro mundial de *Web Services* que permite que aplicações cliente possam de maneira dinâmica encontrar *Web Services* específicos para utilização.

Capítulo 3

Ferramentas para Visualização de Dados Genômicos

A ferramenta utilizada neste trabalho para a visualização de dados genômicos foi o Navegador Genômico Genérico - GBrowse [27]. O GBrowse é um aplicativo *Web* que incorpora uma série de recursos para visualização, ampliação e busca de informações genômicas armazenadas em bancos de dados ou arquivos, montados de acordo com modelos pré-definidos.

O objetivo deste capítulo é apresentar a ferramenta GBrowse e estudar sua utilização na visualização de dados de um projeto genoma. A Seção 3.1 descreve algumas ferramentas de visualização existentes e justifica a escolha do GBrowse. A Seção 3.2 apresenta o projeto GMOD - Modelo Genérico de Banco de Dados de Organismo, do qual o GBrowse faz parte. A Seção 3.3 mostra uma visão geral da ferramenta GBrowse e suas principais características. Esta seção também mostra os formatos de arquivos FASTA e GFF, usados na criação dos bancos de dados a serem visualizados. Na Seção 3.4 descreveremos os tipos de bancos de dados suportados pelo GBrowse e detalhes dos arquivos de configuração destes bancos de dados. Veremos nesta seção como podem ser configuradas as faixas de exibições de dados, que são as regiões no GBrowse onde os dados genômicos são exibidos. Na Seção 3.5 descreveremos o mecanismo de busca de informações do GBrowse. Finalmente na Seção 3.6, apresentaremos diferentes tipos de visualização de dados suportados pela ferramenta.

3.1 Ferramentas de Visualização Existentes

Atualmente existe uma grande quantidade de ferramentas de Bioinformática para visualização de dados genômicos de diversos tipos, como dados resultantes de comparações entre seqüências, genes envolvidos na codificação de proteínas, molduras

de leitura e microarranjo de cDNA entre outros.

Estas ferramentas são construídas em diferentes linguagens de programação como *C*, *Perl* e *Java*, possibilitando o uso em diferentes plataformas como Unix, Linux, Macintosh e Windows.

Quanto à arquitetura, estas aplicações podem ser divididas basicamente em dois grupos: 1) *stand-alone*, ou aplicações que normalmente não dependem de recursos externos como por exemplo banco de dados e servidores, e 2) *Web*, que requerem uma infra-estrutura de *Internet* contendo no mínimo um servidor de páginas com suporte a interpretação de scripts PHP[1] e JSP[2] entre outros.

Algumas aplicações *Web*, como o GBrowse, podem ser instaladas localmente, já outras como o EnteriX, só permitem o uso através da *Internet*. A Tabela 3.1 mostra o resumo das ferramentas levantadas.

Ferramenta	**Arquitetura**	**Descrição**
GBrowse [27]	*Perl*, *Web* (local) e multiplataforma	Ferramenta para visualização de diversos tipos de dados genômicos, como alinhamentos, seqüências e codificação de proteína entre outros
Apollo [34]	*Java*, *stand-alone* e multiplataforma	Ferramenta para anotação e visualização de seqüências genômicas
GenomeComp [64]	*Perl*, *stand-alone* e multiplataforma	Visualização de comparações entre seqüências genômicas através da saída gerada pelo BLAST
Alternative Alignment Visualization Tool - AltAVisT [37]	*Web*	Programa para visualização da comparação de alinhamentos múltiplos de um conjunto de seqüências
EnteriX [24]	*Web* (remoto)	Ferramentas para visualização de genoma bacterial.
Alfresco [48]	*stand-alone* e *Web* (remoto)	Ferramenta para análise de comparação genômica
Vista [26]	*C++* e *Java*, *Web*, multiplataforma	Pacote de ferramentas de programas e bancos de dados para análise comparativa de seqüências genômicas

Tabela 3.1: Ferramentas de Bioinformática para visualização de dados genômicos.

Escolha da Ferramenta

Entre as ferramentas levantadas, optamos pelo GBrowse, que possui características bastante interessantes, descritas abaixo:

- A ferramenta faz parte de um projeto maior em desenvolvimento, do qual atualmente fazem parte mais de vinte ferramentas de Bioinformática de diversos tipos e funcionalidades. Estas ferramentas são integradas através de um esquema de banco de dados comum a todas;

- Suporta o esquema de banco de dados genérico do projeto GMOD[28], que

[1] Linguagem de Script embutida em páginas HTML.

[2] *JavaServer Pages* é uma tecnologia *Java* que permite gerar páginas dinâmicas.

é uma iniciativa que busca desenvolver e disponibilizar um modelo de dados comum a várias aplicações;

- Apresenta uma interface amigável, é multiplataforma, trabalha com bancos de dados de uso livre e suporta o formato GFF *(General Feature Format)*[48], que além de ser de fácil uso e interpretação, pode ser utilizado para a troca de dados entre aplicações;

- É uma ferramenta *Web*, o que permite que seus dados sejam facilmente publicados através da *Internet* com um mínimo de esforço de adaptação;

- Não é necessária a instalação da ferramenta em cada máquina onde desejamos utilizá-la. Basta apenas usar um navegador de páginas *Web*, que é normalmente encontrado em grande parte dos sistemas operacionais;

- É uma ferramenta expansível, que por meio de *plugins*[3], permite que novas funcionalidades sejam introduzidas na ferramenta.

3.2 GMOD - Modelo Genérico de Banco de Dados de Organismo

O GMOD - Modelo Genérico de Banco de Dados de Organismo [28] é um projeto de código aberto criado como o objetivo de desenvolver um conjunto completo de programas para criação e administração de um banco de dados de um organismo.

O projeto foi fundado pelo Instituto Nacional de Saúde - NIH e pelo Serviço de Pesquisa Agrícola - USDA, ambos dos Estados Unidos, com a participação de outras instituições com projetos tais como WormBase [61], FlyBase [25], Mouse Genome Informatics [36], Gramene [29], EcoCyc [22] e Saccharomyces Genome Database [50].

Entre os programas desenvolvidos estão ferramentas para visualização e edição de genomas, um modelo para banco de dados relacional e ferramentas para ontologia[4]. A Tabela 3.2 apresenta vários projetos que fazem parte do GMOD.

O principal subprojeto do GMOD é o Chado, cujo esquema representa genericamente um modelo de dados de um organismo genômico. Basicamente todos os demais softwares pertencentes ao GMOD suportam o uso deste esquema, que com todos os seus 9 módulos possui mais de uma centena de tabelas e visões.

[3]Componente de software que agrega recurso a uma aplicação.
[4]Especificação explícita de uma conceituação.

Categoria	Projeto	Status	Versão
Visualização e Anotação de Genoma	Apollo	Lançado	1.5.4
	GBrowse	Lançado	1.62
Processamento e Gerenciamento de Dados	Modelo Chado	Lançado	gmod-0.003
	BioPipe	Beta	Indisponível
Mapeamento Genético	CMap	Lançado	0.14
	MGD mapa de homologia comparativa	Em projeto	Indisponível
Expressão gênica	Java TreeView	Lançado	1.01
	Pathway Tools	Lançado	9.5
Visualização e Anotação de Vias metabólicas	Pathway Tools	Lançado	9.5
Acesso a Dados/*Web*	BioMart	Lançado	0.3
	LuceGene	Lançado	1.4
Ontologias Biológicas	Sequence Ontology	Alfa	1.5
	GOView	Lançado	0.01

Tabela 3.2: Exemplos de projetos que compõem o GMOD, agrupados por categoria.

Neste trabalho não utilizamos o esquema Chado como banco de dados para o GBrowse. Optamos pelo uso de outros bancos de dados gerados através dos arquivos GFF e FASTA, pois são mais simples de serem criados e utilizados.

3.3 GBrowse: Visão Geral

O GBrowse é um software de código livre, desenvolvido em *Perl*[5] e faz uso de bibliotecas do projeto *BioPerl* [11]. O projeto internacional *BioPerl* teve início em 1995, composto por vários desenvolvedores que visavam construir ferramentas em *Perl* de código livre, para uso em Bioinformática e pesquisas genômicas.

O aplicativo possui código livre e integra o projeto GMOD. Sua execução é feita sobre um servidor HTTP[6] Apache [6]. O software possui basicamente as seguintes características:

- Visualização simultânea do genoma em modo detalhado e geral;
- Rolagem e ampliação de regiões selecionadas;
- Inclusão de URLs para outras anotações;
- Personalização da ordem de exibição das *faixas de dados*, que são as regiões onde os dados genômicos são exibidos;
- Pesquisa de dados pela referência, nome ou comentário;

[5]Linguagem de programação de código livre multi-plataforma [43].
[6]Protocolo de Transferência de Hipertexto [59].

- Suporte a anotações usando o formato GFF;
- Suporte a vários tipos de banco de dados.

A Figura 3.1 mostra uma visão geral do GBrowse. Na parte superior é possível fazer buscas por uma determinada característica e os detalhes são exibidos na parte central da tela. Na parte inferior estão localizadas as configurações de exibição dos dados, como seleção das faixas, tamanho das imagens e posicionamento de texto entre outros [54].

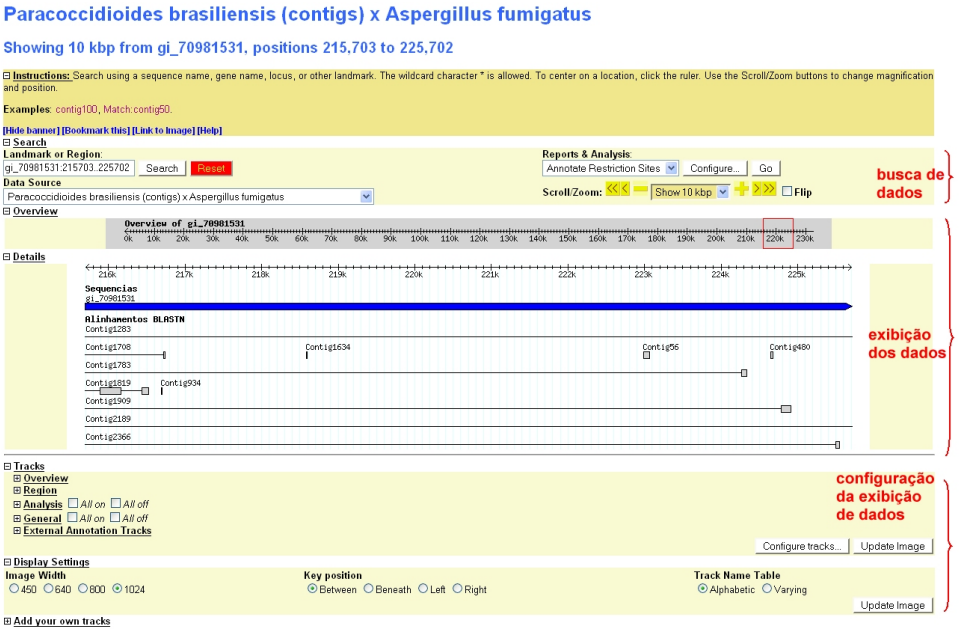


Figura 3.1: Tela principal do navegador genômico genérico - GBrowse [54].

3.3.1 O Formato FASTA

Arquivos no formato FASTA serão utilizados no processo de geração dos bancos de dados utilizados pelo GBrowse e serão descritos a seguir.

O formato FASTA é utilizado para representar uma ou mais seqüências biológicas de ácidos nucléicos ou aminoácidos e pode ser facilmente criado através de editores de texto.

Em um arquivo FASTA, cada seqüência é precedida pelo caractere (>) seguido do nome e descrição da seqüência. Esta descrição tem formato livre, podendo ser usada como desejado. As linhas subseqüentes contêm a seqüência (Figura 3.2).

```
>seq1 Esta é a descrição da primeira seqüência.
AGTACGTAGTAGCTGCTGCTACGTGCGCTAGCTAGTACGTCAC
GACGTAGATGCTAGCTGACTCGATGC

>seq2 Esta é a descrição da segunda seqüência.
CGATCGATCGTACGTCGACTGATCGTAGCTACGTCGTACGTAG
CATCGTCAGTTACTGCATGCTCG
```

Figura 3.2: Exemplo de um arquivo FASTA com duas seqüências de ácidos nucléicos: seq1 e seq2.

3.3.2 O Formato GFF

Arquivos no formato GFF também serão utilizados no processo de geração dos bancos de dados utilizados pelo GBrowse e serão descritos a seguir.

O *General Feature Format* ou *Gene Feature Format* - GFF é um formato de arquivo usado para descrever genes e características associadas como: DNA, RNA e proteínas. O GFF foi especificado pelo Sanger Institute [48].

Este formato foi proposto como um protocolo para troca de informações genéticas. Assim, um programa *A* pode carregar informações de fontes externas geradas por um programa *B*, permitindo o reaproveitamento destas informações. Os programas que ainda não fornecem suporte ao formato GFF podem facilmente fazê-lo devido a simplicidade do formato.

Durante o desenvolvimento do GFF, buscou-se criar um formato que fosse de fácil análise e processamento, para ser utilizado em diferentes linguagens de programação. Assim, ferramentas *Unix* como *grep* e *sort* e scripts *Perl* poderiam facilmente extrair informações de um arquivo GFF.

O Sanger Institute não visava usar o formato GFF para construir uma fonte completa para análise e anotação de seqüência genômica, pois outros formatos e sistemas eram mais apropriados, como o AceDB[7] e o Genotator[8]. As principais desvantagens dos formatos usados por estes sistemas e de outros formatos como o ASN.1 (Abstract Syntax Notation number One [21]), são a necessidade de maior processamento e uma baixa padronização da informação.

O formato GFF proposto possui uma estrutura onde cada característica é descrita em uma linha do arquivo e a ordem das linhas não é relevante. Atualmente, a especificação do GFF encontra-se na sua segunda versão, permitindo a inclusão de informações de RNA, proteínas e DNA.

A Figura 3.3 mostra um exemplo de arquivo GFF na versão 2, onde cada

[7]Banco de dados genômico projetado para manipular dados de Bioinformática [1].
[8]Ferramenta para anotação e navegação de seqüências [30].

linha está dividida em nove colunas. Cada coluna é separada por uma tabulação e os três primeiros campos, que são obrigatórios, não devem possuir valores com espaços. O caractere ponto (.) é usado nas colunas onde um valor não se aplica. Por convenção, um arquivo GFF deve possuir a extensão ".gff".

Colunas								
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
ctgA	fonteA	contig	1	400	.	.	.	Contig ctgA
seq1	fonteA	sequence	1	200	.	+	.	Sequence seq2
seq2	fonteA	sequence	201	400	.	+	.	Sequence seq3
ctgA	fonteA	match	1	201	0.40	+	.	Match seq1
ctgA	fonteA	hsp	1	95	0.40	+	.	Match seq1
ctgA	fonteA	hsp	150	201	0.57	-	.	Match seq1

Figura 3.3: Exemplo de um arquivo GFF.

A seguir veremos mais detalhes de cada uma das colunas de um arquivo GFF:

Coluna 1 - seqüência referência: Nome da referência da seqüência. Com este atributo, é possível diferenciar no mesmo arquivo, diferentes conjuntos de seqüências. Normalmente este atributo será o identificador da seqüência no arquivo FASTA correspondente. Esta coluna é de preenchimento obrigatório.

Coluna 2 - fonte: Fonte da informação. Normalmente indica o programa que gerou a informação, o banco de dados de onde ela foi retirada ou se o dado informado é experimental entre outros. Esta coluna é de preenchimento obrigatório.

Coluna 3 - característica: O nome da característica que está sendo descrita na linha. Qualquer nome pode ser usado. Os nomes mais comuns são: *gene, sequence, repeat,* exon *e CDS*. Esta coluna é de preenchimento obrigatório.

Coluna 4 - início: Valor inteiro indicando o início da seqüência.

Coluna 5 - fim: Valor inteiro indicando o fim da seqüência.

Coluna 6 - escore: Valor real, indicando o escore da característica informada. Usa-se o caractere ponto quando uma característica não possui um escore.

Coluna 7 - cadeia: Indica a direção da característica ou seqüência, assumindo os valores + (cadeia), - (complemento reverso) ou o caractere ponto (.). O caractere ponto é usado quando esta informação não for relevante ou não se aplicar.

Coluna 8 - moldura de leitura: Pode assumir os valores 0, 1, 2 ou o caractere ponto. Em nucleotídeos, o valor 0 indica que a região especificada é uma moldura, onde a primeira base da região corresponde à primeira base de um *códon*; o valor 1 indica que existe uma base extra e a segunda base da região corresponde à primeira base de um *códon*; o valor 2 significa que a terceira base da região é a primeira base de um *códon*. Para proteínas este campo é deixado em branco.

Coluna 9 - atributo: Este campo é usado para descrever atributos adicionais, que devem estar no formato *Classe nome*, onde cada conjunto *Classe nome* é separado por ponto e vírgula (;). Não existe uma formalização para a semântica desta coluna.

3.4 Geração do Banco de Dados Genômico

Para que o GBrowse exiba as informações genômicas em sua interface, precisamos antes gerar um banco de dados contendo estas informações. Para isto são necessários um arquivo no formato FASTA e outro no formato GFF [48] contendo os dados do projeto genômico desejado. O arquivo FASTA contém as seqüências biológicas e o arquivo GFF os demais dados genômicos, como o mapeamento das seqüências, *contigs*, *matchs* ou anotações.

Antes de gerarmos o banco de dados, devemos definir o seu tipo. Existem basicamente três tipos de bancos de dados: banco de dados FASTA e GFF, banco de dados Berkeley e banco de dados relacional. Descreveremos brevemente cada um deles.

Cada tipo de banco de dados deve ser escolhido de acordo com a quantidade de informação a ser exibida e as características do ambiente de execução.

Banco de Dados FASTA e GFF

Este tipo é formado basicamente por um diretório contendo um ou mais arquivos FASTA e GFF, nomeados com as extensões *.fa* e *.gff* respectivamente.

O arquivo GFF será carregado na memória assim que os dados forem requisitados pela primeira vez. O arquivo FASTA será acessado do disco e um índice será automaticamente criado pelo GBrowse para acelerar o acesso ao arquivo.

Devido à carga do arquivo GFF na memória, este tipo de banco de dados se adapta melhor quando temos pequena quantidade de dados, que em média é de até 20.000 linhas, dependendo do total de memória disponível no computador.

Banco de Dados Berkeley

Um banco de dados Berkeley é uma estrutura de armazenamento não relacional. Este tipo de banco de dados é gerado através de um programa fornecido pelo *BioPerl*, que tem como entradas os arquivos FASTA e GFF.

O banco de dados Berkeley tem um bom desempenho e pode comportar um arquivo GFF com até dez milhões de linhas.

Banco de Dados Relacional

Podemos também utilizar um banco de dados relacional para armazenar os dados do projeto. Este tipo possui desempenho e capacidade de armazenamento equivalentes ao banco de dados Berkeley, porém é mais indicado para cenários onde temos vários usuários acessando simultaneamente o sistema e quando se deseja maior recuperação a falhas.

A utilização do banco de dados MySql[9] tem sido exaustivamente testada pela equipe do GBrowse e é recomendada para ambientes de produção.

A Figura 3.4 mostra um esquema do GBrowse com seus principais componentes: o arquivo de configuração do projeto e os três tipos de banco de dados suportados.

Realizamos um teste para medir o desempenho de cada um destes três tipos de bancos de dados suportados pela ferramenta GBrowse. O teste consistiu em medir o tempo gasto pela ferramenta para mostrar as bases de uma seqüencia com 2.671.808 nucleotídeos. Foram preparados três bancos de dados para cada um dos três tipos (Tabela 3.3). Cada banco de dados continha dados comparativos entre o *Paracoccidioides brasiliensis* e o *Aspergillus fumigatus*.

Número de registros	**Banco de dados**		
52.000	**Memória**	**Berkeley**	**Relacional**
104.000	**Memória**	**Berkeley**	**Relacional**
147.000	**Memória**	**Berkeley**	**Relacional**

Tabela 3.3: Os nove bancos de dados gerados para o teste de desempenho.

A Figura 3.5 mostra os resultados do teste de desempenho entre os tipos de bancos de dados suportados pela ferramenta GBrowse. Como podemos notar, o tipo de banco de dados relacional possui o melhor desempenho, seguido de perto

[9]Banco de dados relacional multi-plataforma [38].

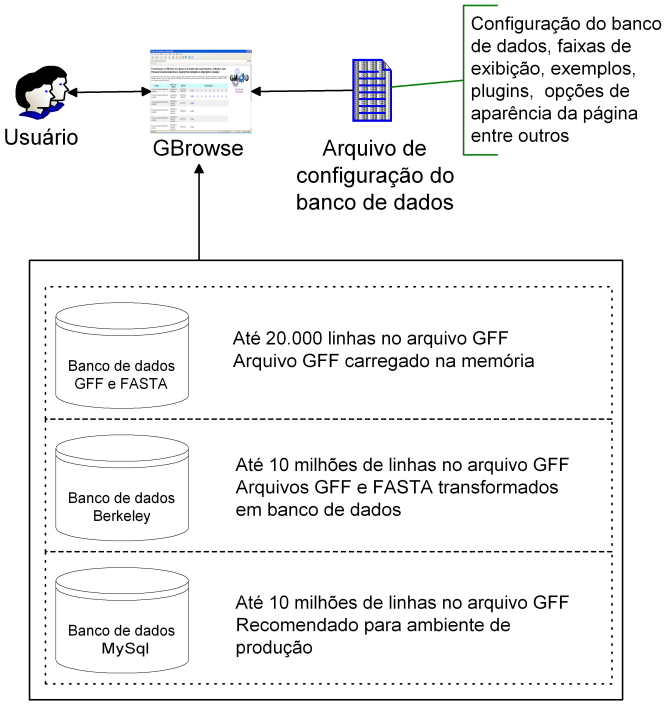


Figura 3.4: Esquema do GBrowse com seus principais componentes.

pelo tipo de banco de dados Berkeley. Por último, com um desempenho bastante inferior, temos o tipo de banco de dados em memória. O teste foi realizado sobre um PC com processador Intel® Centrino® 1.6 MHz, 1 Giga byte de memória RAM, executando o GBrowse versão 1.64 sobre o sistema operacional Windows XP®.

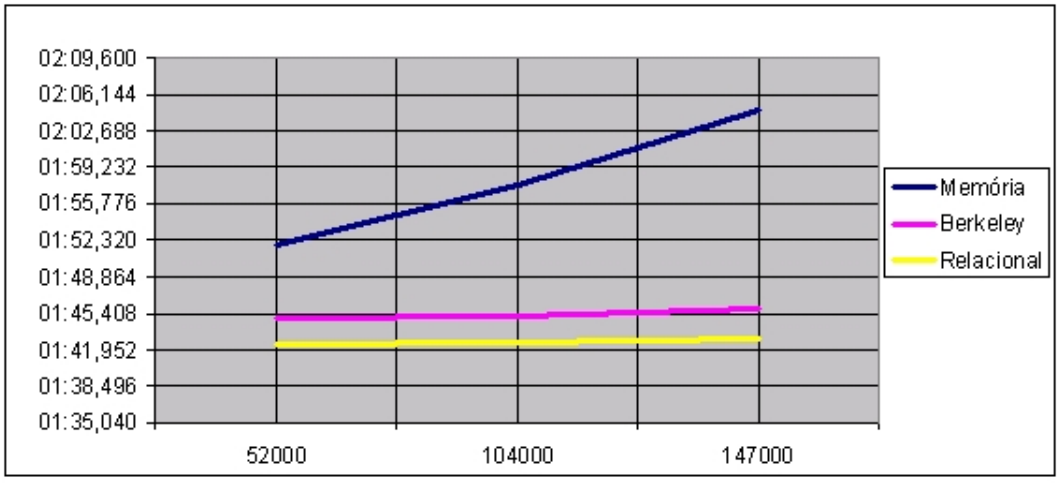


Figura 3.5: Gráfico comparativo do teste de desempenho entre os tipos de bancos de dados suportados pelo GBrowse.

Tipo de banco de dados	**Número de registros**		
	52.000	104.000	147.000
Memória	01:51,941	01:57,689	02:04,685
Berkeley	01:44,861	01:45,201	01:45,792
Relacional	01:42,438	01:42,638	01:43,018

Tabela 3.4: Resultado do teste de desempenho entre os tipos de bancos de dados suportados pelo GBrowse.

3.4.1 Configuração do Banco de Dados

Como mencionado anteriormente, para a montagem de um banco de dados a ser visualizado no GBrowse, deve ser criado um arquivo FASTA, contendo as seqüências genômicas e um arquivo GFF, contendo as demais características a serem exibidas, como informações das seqüências e os alinhamentos entre outras.

O último passo necessário para que o GBrowse possa exibir o banco de dados é a criação do arquivo de configuração deste banco de dados, onde são feitos ajustes que refletem desde o modo como o GBrowse acessa as informações até as cores e formas de exibição dos dados. A Tabela 3.5 mostra de forma resumida alguns ítens que podem ser ajustados no arquivo de configuração, onde os principais estão em destaque. Um exemplo completo do arquivo de configuração do banco de dados encontra-se no apêndice B.

Item	**Descrição**	**Exemplo(s)**
description	Título que identifique o projeto	*Pb x Aspergillus fumigatus*
dbadaptor	Adaptador *Perl* que interpretará os dados	Bio::DB::GFF
dbargs	Configuração do banco de dados, informando qual o adaptador e suas configurações	-adaptor berkeleydb *(banco de dados Berkeley)* -dsn '/databases/pbaf_bdb' *(localização do banco de dados)*
aggregators	Agregadores de dados usados no projeto	match *(agrega HSPs do BLAST)*
plugins	Módulos adicionais incorporados ao GBrowse, com objetivo de estender suas funcionalidades	Aligner *(permite visualizar o alinhamento múltiplo da região selecionada)*
default features	Lista de faixas de exibição de dados que estarão pré-selecionadas	Sequence Alignments
reference class	Classe dos objetos usados para estabelecer as coordenadas de referência	ContigA
examples	Exemplos a serem mostrados como sugestões de pesquisa	ContigA Sequence:gi_70982280:1..1000
automatic classes	Classes que serão automaticamente usadas quando for informado um identificador sem a classe	Sequence
default segment	Tamanho do segmento a ser exibido, em número de bases	50000
zoom levels	Níveis de ampliação, em número de bases	100 200 1000 5000 50000
tracks	Faixas de visualização. Cada faixa possui um conjunto de parâmetros próprios	[Sequences], [Alignments], [Translation]

Tabela 3.5: Alguns ítens contidos no arquivo de configuração do banco de dados exibido pelo GBrowse, onde os principais ítens são mostrados em destaque.

As **Faixas de Visualização** de dados estão contidas no arquivo de configuração. Cada faixa de visualização configurada possibilita a visualização das respectivas informações numa região do GBrowse e pode-se utilizar estas faixas para exibir informações de seqüências, alinhamentos ou codificação de proteínas entre outras.

A Figura 3.6 mostra um arquivo GFF e sua respectiva configuração da faixa de visualização para exibir as seqüências genômicas. Neste exemplo, a faixa de visualização chama-se "FaixaSequencias"e informa:

- ***feature***: coluna referência dos dados no arquivo GFF;
- ***glyph***: a imagem a ser exibida para representar as informações;
- ***stranded***: 1 indica que deve ser considerada a direção (cadeia ou complemento reverso) na imagem exibida;
- ***bgcolor***: a cor de fundo;
- ***height***: a altura das imagens;
- ***key***: a descrição da faixa.

ctgA	exemplo	contig	1	50000	.	.	Contig ctgA
ctgA	exemplo	seq	1659	1984	+	.	Sequencia s01
ctgA	exemplo	seq	3014	6130	+	.	Sequencia s02
ctgA	exemplo	seq	4715	5968	-	.	Sequencia s03
ctgA	exemplo	seq	13280	16394	+	.	Sequencia s04

(a)

```
[FaixaSequencias]
feature = seq
glyph = generic
stranded = 1
bgcolor = blue
height = 10
key = Exemplo de Seqüências
```

(b)

Figura 3.6: (a) Exemplo de um arquivo GFF descrevendo um *contig* e as seqüências que o formam. (b) A respectiva configuração da faixa de visualização dos dados deste arquivo GFF.

Cada visualização que será mostrada na Seção 3.6 possui sua respectiva *faixa*

de visualização devidamente configurada neste arquivo. Outros exemplos de faixa de visualização podem ser vistos no apêndice B.

3.5 Busca de Dados

Após a configuração e geração do banco de dados, pode-se utilizar o GBrowse para a procura e visualização dos dados do projeto genoma.

O GBrowse possui um mecanismo de busca bastante flexível. Para demonstrar esta flexibilidade utilizaremos o exemplo da Figura 3.6. Uma informação pode ser localizada por sua referência ou nome, conforme veremos a seguir:

- **Busca pela seqüência referência**

  No exemplo fornecido, se desejarmos localizar uma referência chamada "ctgA", o programa irá mostrar o *contig* inteiro. Quando desejarmos mostrar uma determinada região dentro deste *contig*, podemos usar "ctgA:5000..8000", onde 5000 e 8000 são respectivamente o início e o fim da seqüência dentro deste *contig*.

- **Busca pelo nome da característica**

  Podemos também localizar uma informação pelo nome do objeto, que deve estar precedido pela sua classe, no formato *Classe nome*. Na Figura 3.6, em *Sequencia s01* e *Sequencia s02*, *Seqüência* é a classe e *s01* e *s02* são os nomes das características.

  Em algumas buscas é possível utilizar o caractere * nos casos em que fornecemos parcialmente o nome da característica a ser localizada. Este caractere pode ser inserido em qualquer parte do texto, como veremos a seguir: s0*, s*2, *03.

3.6 Visualização de Dados

O GBrowse é uma ferramenta poderosa para visualização, que gera imagens em tempo real. De acordo com as informações disponíveis em cada banco de dados pode-se configurar diferentes faixas de visualização. A seguir descreveremos algumas faixas.

3.6.1 Seqüências

Cada seqüência de nucleotídeos ou aminoácidos contida no arquivo FASTA pode ser visualizada graficamente no GBrowse, conforme exemplificado no trecho mostrado no arquivo GFF da Figura 3.7.

ctgA	exemplo	contig	1	50000	.	.	.	Contig ctgA
ctgA	exemplo	sequencia	44705	47713	.	-	.	Sequencia s01
ctgA	exemplo	sequencia	24562	28338	.	+	.	Sequencia s02
ctgA	exemplo	sequencia	36649	40440	.	-	.	Sequencia s03
ctgA	exemplo	sequencia	37242	38653	.	+	.	Sequencia s04
ctgA	exemplo	sequencia	36034	38167	.	+	.	Sequencia s05

Figura 3.7: Trecho do arquivo GFF usado na visualização das seqüências.

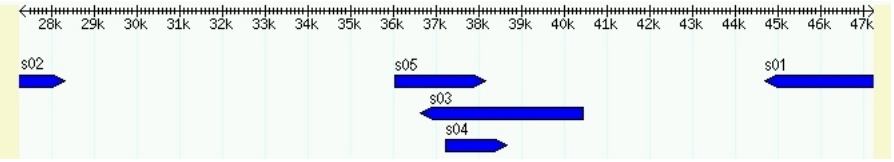


Figura 3.8: Faixa de visualização de seqüências de nucleotídeos ou aminoácidos do arquivo GFF mostrado na Figura 3.7.

Na Figura 3.8 temos a visualização das informações contidas neste arquivo, onde notamos as cadeias, complementos reversos e o tamanho e localização de cada uma destas seqüências. Ao selecionamos uma determinada seqüência, podemos ver detalhes (Figura 3.9) como as bases que a compõem, o tamanho e a classe entre outros.

Sequencia:s02 Details

Name:	**s02**
Class:	Sequencia
Type:	Sequencia
Source:	exemplo
Position:	ctgA:24562..28338 (+ strand)
Length:	3777

```
>s02 class=Sequencia position=ctgA:24562..28338 (+ strand)
ctgcctaccgggtcgaattatttacgcgtgttacaatatgtaatttagaaaaagggattgctggtcgatgcgtctccaag
ggatttttttatctaaaagcatccttttgggtgtactctgatcgcacgtcgcagacagcagtgggttttgacgcagtccgt
aggcccacagactcgtttgttgtttattaatcccaggggagcgttgaagccacacctattctgtagctgtttgaaaggta
gctagcccggatattactcaaggtgactcccttcagaatcacacgtcgctggagtcgccacagggtggcatatacgagtg
atagagcaccttacttttcgaggtagcggtacattagtgcaacgatgaacccactatagtcttagtgatttcatgttttac
ttacgcgaaaacgtggggttttgtcaacacgtatacgttgaatgcacatgcctcatcctaaactgatgcactgccacaag
tctgaaagagcgacagtctgcaacatagcggaaggttacgcccaagccagtggtgatcccccataagcttggagggactc
cccttagcgttggatgtctttgccccagcggcctcggtgtacgggttctccaccccactatggtttggaactatgaagag
gtacggcaacctacccgaggcaccaaatcgtgaacctacgcctatatatacggatagcagggtatccattcttaccatga
gctcgtaaaccactccgctgaattcgatgggctttggcgcacatcaccgtttctatcacagatctgtcaacggaatctaa
cgctatttactcggcgcacacagatcggaaaacccaactgtggcgcgggacggactccaggaatcgttacgcgttatcac
```

Figura 3.9: Detalhes da seqüência s02.

3.6.2 Alinhamentos

É comum em projetos genomas a comparação das seqüências sendo estudadas com uma banco de seqüências conhecidas, através da execução de programas como o BLAST. A saída texto gerada pelo BLAST, contendo os alinhamentos resultantes, pode também ser visualizada no GBrowse. Na Figura 3.10 temos

um trecho do arquivo GFF que originou a saída vista na Figura 3.11. Nesta última figura, podemos observar diferentes segmentos *(seg01, seg02, seg03)*, cada um sendo formado por vários HSPs. Observamos também em que regiões estes segmentos e HSPs ocorrem dentro da seqüência comparada.

ctgA	exemplo	match	6885	8999	.	-	.	Match seg01
ctgA	exemplo	HSP	6885	7241	.	-	.	Match seg01
ctgA	exemplo	HSP	8055	8080	.	-	.	Match seg01
ctgA	exemplo	HSP	8306	8999	.	-	.	Match seg01
ctgA	exemplo	match	5233	6101	.	-	.	Match seg02
ctgA	exemplo	HSP	5233	5302	.	-	.	Match seg02
ctgA	exemplo	HSP	5800	6101	.	-	.	Match seg02
ctgA	exemplo	match	1233	5825	.	-	.	Match seg03
ctgA	exemplo	HSP	1233	1302	.	-	.	Match seg03
ctgA	exemplo	HSP	2442	2854	.	-	.	Match seg03
ctgA	exemplo	HSP	4869	4935	.	-	.	Match seg03
ctgA	exemplo	HSP	5404	5825	.	-	.	Match seg03

Figura 3.10: Trecho do arquivo GFF usado na visualização de alinhamentos produzidos pelo BLAST.

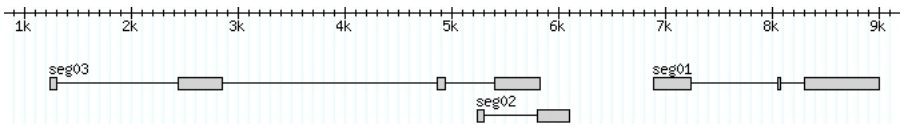


Figura 3.11: Faixa de visualização de alinhamentos BLAST do arquivo GFF mostrado na Figura 3.10, onde cada retângulo representa um HSP.

3.6.3 Genes que Codificam Proteínas

O GBrowse pode mostrar o gene que codifica proteína. Para isto devem ser definidos os elementos gene, mRNA, CDS, 5' UTR e 3' UTR (Figura 3.12).

O resultado do trecho mostrado na figura anterior pode ser visto na Figura 3.13, onde observamos um gene chamado EDEN e dois mRNAs: EDEN.1 e EDEN.2. A proteína codificada é a *Kinase* e cada região (retângulos e retângulos com ponta triangular, indicando a direção) pode eventualmente estar dividida em CDS e UTR. Em EDEN.1 por exemplo, o primeiro e quarto segmentos possuem regiões 5'-UTR e 3'-UTR respectivamente. Observamos ainda que podem ser identificadas as regiões na seqüência nas quais cada parte envolvida na codificação da proteína ocorre.

3.6.4 Molduras de Leitura *(Reading Frames)*

Para visualizarmos as molduras de leitura é necessária apenas a configuração da faixa de exibição, já que as mesmas serão automaticamente geradas de acordo

ctgA	exemplo	gene	1050	9000	.	+	.	Gene EDEN;Note "protein kinase"
ctgA	exemplo	mRNA	1050	9000	.	+	.	mRNA EDEN.1;Gene EDEN
ctgA	exemplo	5'-UTR	1050	1200	.	+	.	mRNA EDEN.1
ctgA	exemplo	CDS	1201	1500	.	+	0	mRNA EDEN.1
ctgA	exemplo	CDS	3000	3902	.	+	0	mRNA EDEN.1
ctgA	exemplo	CDS	5000	5500	.	+	0	mRNA EDEN.1
ctgA	exemplo	CDS	7000	7608	.	+	0	mRNA EDEN.1
ctgA	exemplo	3'-UTR	7609	9000	.	+	.	mRNA EDEN.1
ctgA	exemplo	mRNA	1050	9000	.	+	.	mRNA EDEN.2;Gene EDEN
ctgA	exemplo	5'-UTR	1050	1200	.	+	.	mRNA EDEN.2
ctgA	exemplo	CDS	1201	1500	.	+	0	mRNA EDEN.2
ctgA	exemplo	CDS	5000	5500	.	+	0	mRNA EDEN.2
ctgA	exemplo	CDS	7000	7608	.	+	0	mRNA EDEN.2
ctgA	exemplo	3'-UTR	7609	9000	.	+	.	mRNA EDEN.2

Figura 3.12: Trecho do arquivo GFF usado na visualização da codificação de proteína.

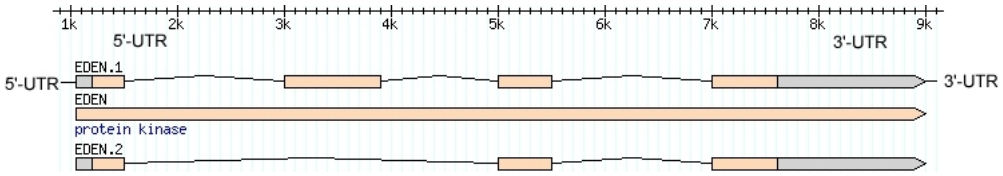


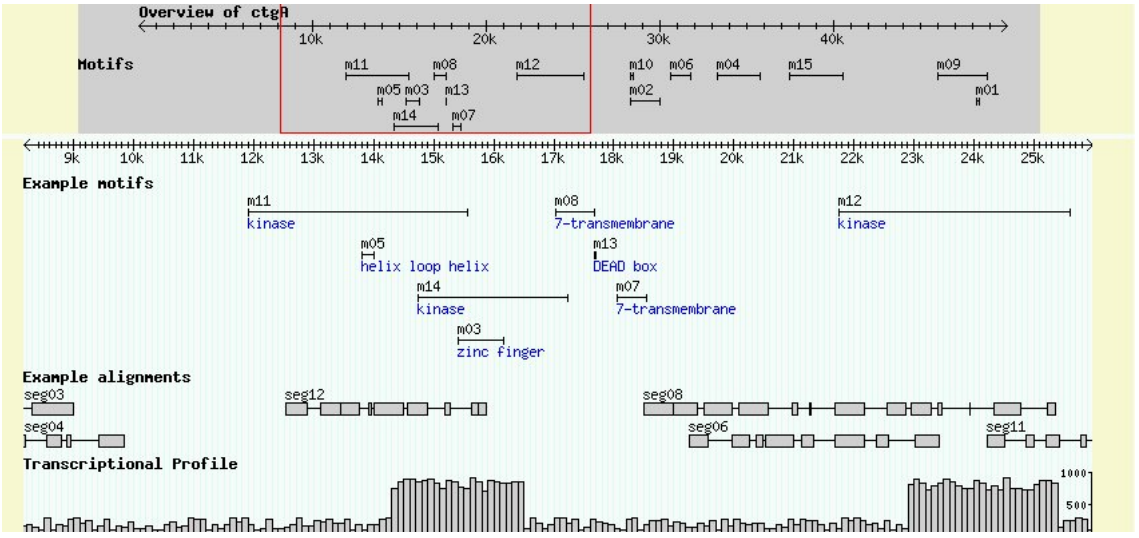
Figura 3.13: Faixa de visualização de CDS e UTR do arquivo GFF da Figura 3.12.

com a seleção de uma região na seqüência. A Figura 3.14 mostra a visualização da faixa de molduras de leitura, onde cada moldura é apresentada em uma linha.

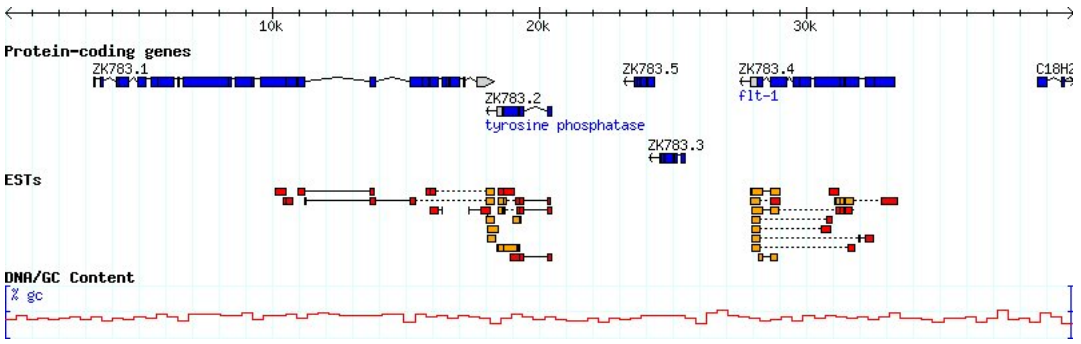
Figura 3.14: Faixa de visualização das molduras de leitura *(Reading Frames)*.

3.6.5 Outras Visualizações

Dependendo do tipo de informação disponível no projeto, o GBrowse permite construir ricas e diferentes faixas de visualização (Figura 3.15), como Motifs, ESTs, perfil da transcrição, codificação de proteínas entre outras.



(a)



(b)

Figura 3.15: Outras faixas de visualização de dados no GBrowse: (a) Motifs, alinhamentos e perfil de transcrição, e (b) codificação de proteínas, EST e conteúdo DNA/GC.

Capítulo 4

Método para Visualização de Dados Genômicos no GBrowse

O uso da ferramenta para visualização de dados genômicos GBrowse pode significar uma grande economia de tempo, se comparado com os recursos necessários para desenvolver uma ferramenta com tais características. Porém, para utilizarmos eficientemente a ferramenta, é interessante definir um método para descrever todos os passos e detalhes necessários para preparar um banco de dados a ser usado pela ferramenta.

Portanto, neste capítulo apresentamos o método proposto para visualização de dados genômicos no GBrowse. Na Seção 4.1 descrevemos o método proposto, apresentando os pré-requisitos necessários à execução do método e as etapas que o compõem. Na Seção 4.2, realizamos experimentos para demonstrar o método e fazemos uma discussão. Esta seção também apresenta algumas contribuições realizadas para facilitar o uso da ferramenta para os bancos de dados gerados neste trabalho.

4.1 Descrição do Método

O método de visualização de dados genômicos no GBrowse é compreendido em cinco sucessivas etapas:

1. Formatação dos Bancos de Dados BLAST
2. Comparação de Seqüências Através do BLAST
3. Conversão das Saídas BLAST em arquivos GFF
4. Geração dos Bancos de Dados Berkeley

5. Configuração do Projeto e Visualização no GBrowse

As etapas 1 e 2 podem ser criadas de acordo com as necessidades de cada projeto. Particularmente, neste trabalho, estas etapas foram construídas de acordo com as demandas do Laboratório de Biologia Molecular da Universidade de Brasília.

Na etapa 2 utilizamos os programas BLASTN e TBLASTX. O BLASTN foi utilizado porque a seqüência de consulta do *P. brasiliensis* e as seqüências do banco do *A. fumigatus* e *A. nidulans* estavam descritas em nucleotídeos. Já o TBLASTX foi utilizado pois transforma em aminoácidos os nucleotídeos das seqüências de consulta e do banco nas seis fases de leitura. Como um único aminoácido pode ser obtido com vários códons diferentes, as comparações neste caso têm mais chances de encontrar similaridade entre as seqüências.

A Figura 4.1 mostra o esquema geral do método utilizado. O termo *Organismo A*, referenciado nesta figura, significa o organismo utilizado na comparação com *Paraccocidioides brasiliensis*, que é o *Organismo B*.

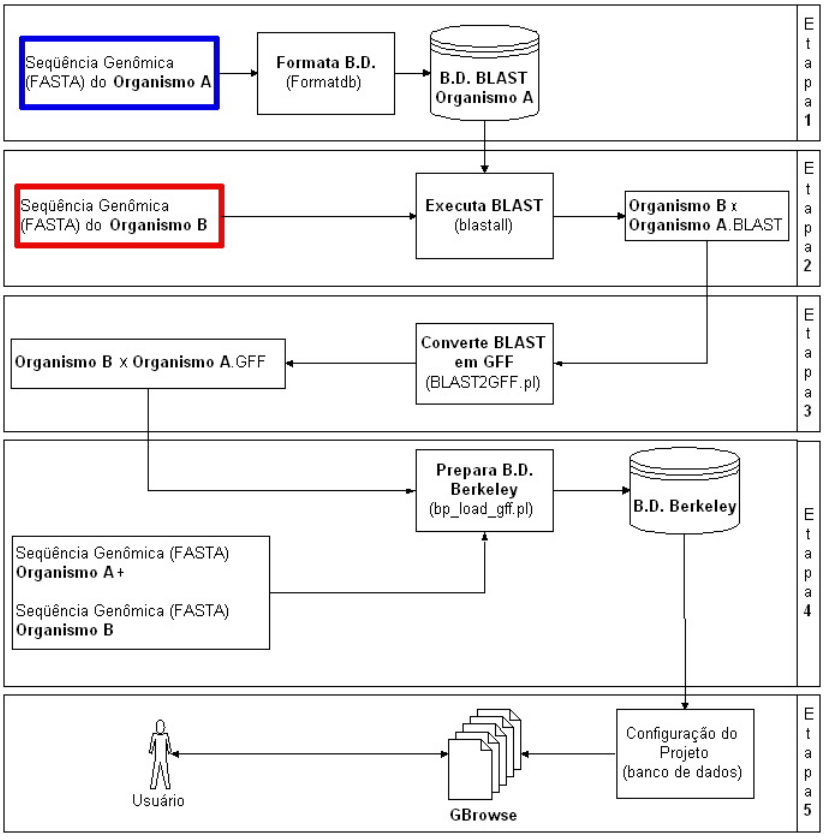


Figura 4.1: Método utilizado para visualização de dados Genômicos no GBrowse.

Pré-requisitos: para a execução deste método, faz-se necessário que os seguintes

programas estejam devidamente instalados e configurados, de acordo com as instruções fornecidas por cada um deles: *Perl*, *Bioperl*, *GBrowse*, *formatdb*, *blastall* e servidor *Apache*. A etapa 1 utiliza o *formatdb*, a etapa 2 utiliza o *blastall* e as etapas 3, 4 e 5 utilizam o *Perl*, *Bioperl* e o *GBrowse*. Cada etapa será descrita a seguir.

Etapa 1: Formatação do Banco de Dados BLAST

A primeira etapa do método é a formatação dos bancos de dados BLAST, realizada através da ferramenta *formatdb* e dos arquivos de seqüências.

O *formatdb* é um programa fornecido com o pacote de softwares do BLAST e pode criar um banco de dados de nucleotídeos ou proteínas.

Para a execução do programa *formatdb*, devemos informar os seguintes parâmetros: **-i**, que é o arquivo de seqüências FASTA, **-n** e **-t**, que são respectivamente o nome e o título para o banco de dados e **-p**, que informa o tipo das seqüências contidas no arquivo FASTA: *F* para seqüências de nucleotídeos e *T* para seqüências de proteínas (aminoácidos). A Figura 4.2 mostra a linha de execução utilizada na formatação do banco de dados.

formatdb ***-i*** arquivo_fasta ***-n*** nome_do_banco ***-t*** título_do_banco ***-p*** F

Exemplo: **formatdb** ***-i*** afu.fasta ***-n*** Aspergillus_fumigatus_nt ***-t*** Aspergillus_fumigatus_nt ***-p*** F

Figura 4.2: Linha de execução do programa *formatdb*.

Etapa 2: Comparação de Seqüências Através do BLAST

O objetivo desta etapa é realizar a comparação entre as seqüências de um organismo e as seqüências contidas nos bancos de dados BLAST criados na etapa anterior. Para isto utilizamos o programa *blastall*, também fornecido junto com o pacote de softwares do BLAST.

O *blastall* aceita diversos parâmetros, destacando-se o parâmetro **-p**, que informa qual programa BLAST deverá ser executado para a comparação das seqüências. Neste experimento utilizamos os programas BLASTN e TBLASTX. Os demais parâmetros são: **-d** é o nome do banco de dados para comparação, **-i** o arquivo FASTA para a comparação e **-o** é o nome do arquivo de saída. A Figura 4.3 mostra o *blastall* e os parâmetros utilizados.

blastall *-p* programa ***-d*** nome_do_banco ***-i*** sequencia_consulta_fasta ***-o*** saida

Exemplo: **blastall *-p*** BLASTN ***-d*** Aspergillus_fumigatus_nt ***-i*** pbcontigs.fasta ***-o*** afu_pbcontigs_blastn.blast

Figura 4.3: Linha de execução do programa *blastall*.

Etapa 3: Conversão das Saídas BLAST em arquivos GFF

Nesta etapa, cada arquivo de saída BLAST será convertido em um arquivo no formato GFF, necessário para a geração do banco de dados para o GBrowse.

Para a conversão, utilizamos um programa *Perl* chamado *blast2gff.pl*, encontrado no pacote da instalação do GBrowse.

O script *blast2gff.pl* possui os seguintes parâmetros: **-blast_result_file** é o nome do arquivo blast que desejamos converter, **–reference_sequence_file** é o nome do arquivo FASTA contendo todas as seqüências referenciadas pelo arquivo BLAST e **–gff_output_file** é o nome do arquivo de saída. A Figura 4.4 mostra a linha de execução do programa *blast2gff*.

blast2gff.pl –***blast_result_file*** arquivo_blast –***reference_sequence_file*** arquivo_fasta –***gff_output_file*** saida

Exemplo: **blast2gff.pl** –***blast_result_file*** afu_pbcontigs_blastn.blast –***reference_sequence_file*** afu_pbcontigs.fasta –***gff_output_file*** afu_pbcontigs_blastn.gff

Figura 4.4: Linha de execução do programa *blast2gff.pl*.

Etapa 4: Geração dos Bancos de Dados Berkeley

Nesta etapa é realizada a geração dos bancos de dados.

Analisando os tipos de banco de dados disponíveis para o GBrowse optamos por utilizar o tipo Berkeley, pois além da facilidade de uso, visto que não requer um sistema gerenciador de banco de dados - SGBD, este banco de dados apresenta um desempenho equivalente a um banco relacional [27].

Para criarmos os bancos de dados Berkeley, utilizamos o script em *Perl*, chamado *bp_load_gff.pl*. Este script faz parte do pacote *Bioperl* e está disponível após sua instalação.

O *bp_load_gff.pl* possui os seguintes parâmetros: **-c** informa ao script que as eventuais informações existentes no banco de dados devem ser removidas, **-a** informa que adaptador será utilizado e **-d** indica o diretório onde será criado o

banco de dados. Os últimos dois parâmetros são os nomes dos arquivos FASTA e GFF. A Figura 4.5 mostra a linha de execução do programa.

bp_load_gff.pl *-c -a* berkeleydb ***-d*** diretório arquivo_fasta arquivo_gff

Exemplo: **bp_load_gff.pl *-c -a*** berkeleydb ***-d*** afu_pbcontigs_blastn_bdb afu_pbcontigs.fasta afu_pbcontigs_blastn.gff

Figura 4.5: Linha de execução do programa *bp_load_gff.pl*.

De acordo com cada projeto, podemos combinar ou separar os arquivos GFF para criarmos diferentes bancos de dados. Esta decisão depende de fatores como o nível desejado de comparação entre os dados e do poder computacional disponível no servidor com o GBrowse, já que arquivos maiores demandam maior processamento, influenciando no tempo de resposta da ferramenta.

Etapa 5: Configuração dos Bancos de Dados e Visualização no GBrowse A última etapa do método é a configuração dos bancos de dados. Basicamente, nesta etapa criamos o arquivo de configuração para que o GBrowse possa reconhecer e assim disponibilizar a visualização das informações contidas nestes bancos de dados.

Devemos criar um arquivo para cada banco de dados, onde informaremos a descrição do projeto, o adaptador e o caminho para o banco de dados, os agregadores de dados e as faixas de visualização das seqüências e alinhamentos entre outros. A configuração mínima requerida pode ser vista na tabela 4.1.

Seção		
[general]	[Sequences]	[Alignments]
description	feature	feature
db_adaptor	glyph	glyph
db_args	stranded	key
aggregators	bgcolor	
reference class	height	
automatic classes	key	

Tabela 4.1: Informações mínimas contidas no arquivo de configuração do banco de dados exibido pelo GBrowse.

O arquivo completo da configuração do banco de dados resultante da comparação das seqüências do *Paraccocidioides brasiliensis (contigs) com o Aspergillus fumigatus* pode ser visto no apêndice B.

4.2 Experimentos e Discussão

Nesta seção apresentamos experimentos para validar o método proposto. O experimento utilizou as seqüências do *Paracoccidioides brasiliensis*, *Aspergillus fumigatus* e *Aspergillus nidulans*, gerando como resultado final uma série de bancos de dados configurados para visualização no GBrowse. De acordo com o método proposto, temos cinco etapas a cumprir. A Figura 4.6 apresenta de forma gráfica todos os experimentos realizados.

Etapa 1: Formatação do Banco de Dados BLAST
Os arquivos FASTA contendo as seqüências de nucleotídeos dos fungos *Aspergillus fumigatus* e *Aspergillus nidulans* foram submetidas ao programa *formatdb* e deram origem aos bancos de dados BLAST (Tabela 4.2).

Organismo	**Arquivo FASTA** (-i)	**Nome** (-n) e **Título do Banco**(-t)	**Tipo** (-p)	**Saída**
Aspergillus fumigatus	afu.fasta	Aspergillus_fumigatus_nt	F	Banco de dados de nucleotídeos com todos os cromossosmos do *Aspergillus fumigatus*
	afu_crom1.fasta	Aspergillus_fumigatus_crom1_nt	F	Banco de dados de nucleotídeos do cromossomo 1 do *Aspergillus fumigatus*
	afu_crom2.fasta	Aspergillus_fumigatus_crom2_nt	F	Banco de dados de nucleotídeos do cromossomo 2 do *Aspergillus fumigatus*
	afu_crom3.fasta	Aspergillus_fumigatus_crom3_nt	F	Banco de dados de nucleotídeos do cromossomo 3 do *Aspergillus fumigatus*
	afu_crom4.fasta	Aspergillus_fumigatus_crom4_nt	F	Banco de dados de nucleotídeos do cromossomo 4 do *Aspergillus fumigatus*
	afu_crom5.fasta	Aspergillus_fumigatus_crom5_nt	F	Banco de dados de nucleotídeos do cromossomo 5 do *Aspergillus fumigatus*
	afu_crom6.fasta	Aspergillus_fumigatus_crom6_nt	F	Banco de dados de nucleotídeos do cromossomo 6 do *Aspergillus fumigatus*
	afu_crom7.fasta	Aspergillus_fumigatus_crom7_nt	F	Banco de dados de nucleotídeos do cromossomo 7 do *Aspergillus fumigatus*
	afu_crom8.fasta	Aspergillus_fumigatus_crom8_nt	F	Banco de dados de nucleotídeos do cromossomo 8 do *Aspergillus fumigatus*
Aspergillus nidulans	ani.fasta	Aspergillus_nidulans_nt	F	Banco de dados de nucleotídeos do *Aspergillus nidulans*

Tabela 4.2: Formatação dos bancos de dados BLAST de nucleotídeos através do programa *formatdb*.

O *Aspergillus fumigatus* possui nove arquivos FASTA, onde um contém as

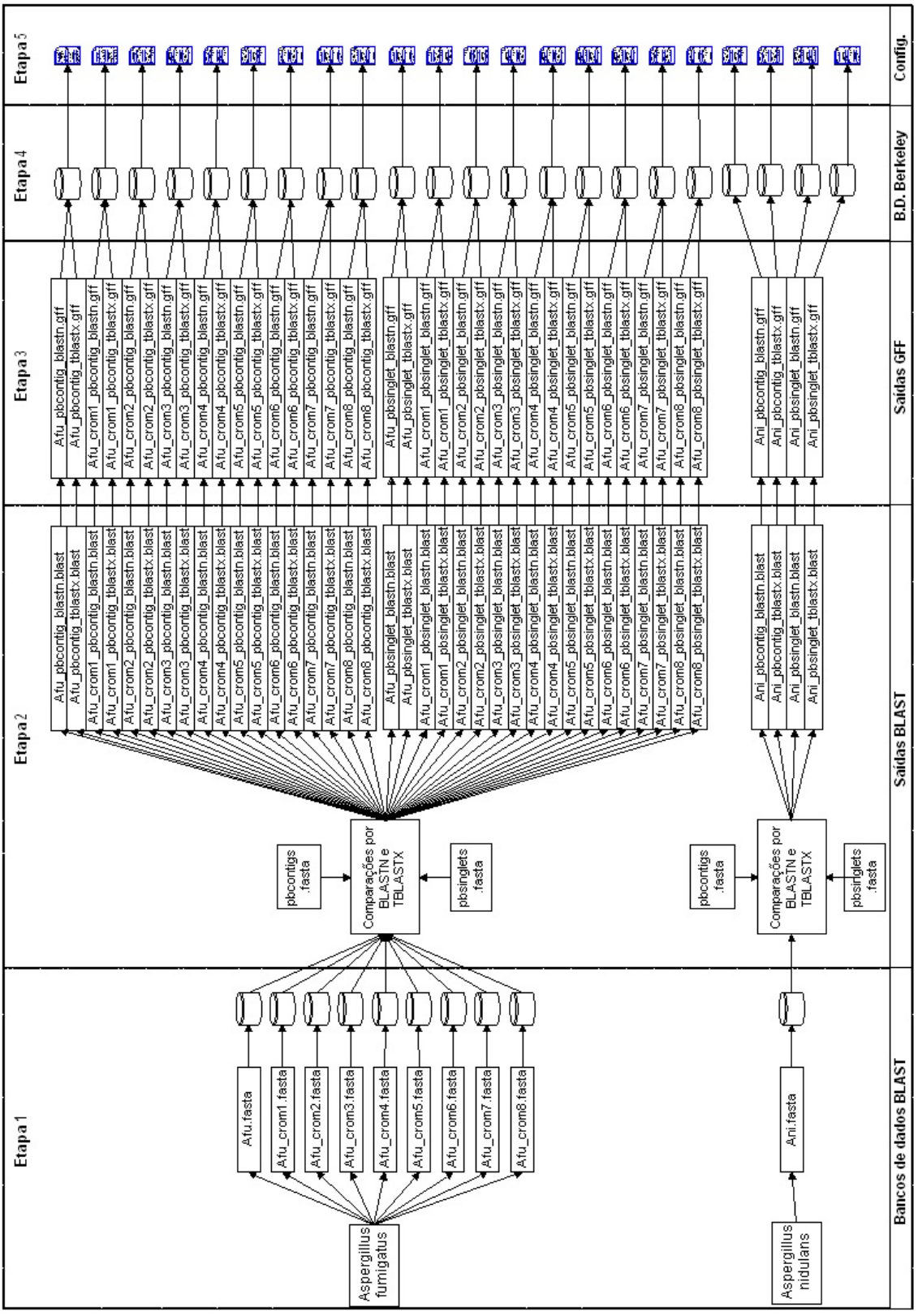


Figura 4.6: Esquema geral dos experimentos realizados utilizando o método de visualização de dados genômicos no GBrowse.

seqüências de todos os seus cromossomos e os demais arquivos contém respectivamente as seqüências de cada um dos seus oito cromossomos. Já o *Aspergillus nidulans* possui somente um arquivo FASTA contendo as seqüências de todos os seus oito cromossomos. Assim, obtivemos respectivamente nove bancos de dados BLAST para *Aspergillus fumigatus* e um banco de dados BLAST para *Aspergillus nidulans*.

Etapa 2: Comparação de Seqüências Através do BLAST
As seqüências do *Paracoccidioides brasiliensis* estão divididas em dois arquivos: um contendo os *contigs* e outro os *singlets*.

Estes *contigs* e *singlets* foram comparados com os bancos de dados BLAST gerados na etapa anterior, utilizando os programas BLASTN e TBLASTX.

As comparações realizadas para *Aspergillus fumigatus* geraram um total de 36 saídas BLAST e para *Aspergillus nidulans* foram geradas apenas 4 saídas (Tabela 4.3).

Etapa 3: Conversão das Saídas BLAST em arquivos GFF
Nesta etapa, tomamos os arquivos de saída BLAST gerados pela etapa anterior e os convertemos em arquivos do formato GFF.

Ao final desta etapa obtivemos 36 arquivos BLAST para *Aspergillus fumigatus* e 4 arquivos para *Aspergillus nidulans* (Tabela 4.4).

Etapa 4: Geração dos Bancos de Dados Berkeley
A geração dos bancos de dados Berkeley é mostrada na Tabela 4.5. Foram 18 bancos de dados para *Aspergillus fumigatus* e 4 para *Aspergillus nidulans*. Como podemos notar nesta tabela, juntamos os arquivos GFFs de BLASTN e TBLASTX do *Aspergillus fumigatus* afim de reduzir o número de banco de dados gerados.

Etapa 5: Configuração dos Bancos de Dados e Visualização no GBrowse
Nesta etapa criamos um arquivo de configuração para cada um dos 22 bancos de dados gerados na etapa anterior. A Figura 4.6 apresenta os ajustes realizados em cada um destes arquivos.

Visualização dos Dados no GBrowse:
A seguir mostraremos algumas visualizações no GBrowse. Para a demonstração utilizamos o banco de dados das seqüências *contigs* do *Paracoccidioides brasi-*

Organismo	**Banco** (-d)	**Programa BLAST** (-p)	**Seqüência consulta FASTA** (-i)	**Saída** (-o)
Aspergillus fumigatus	Aspergillus_fumigatus_nt	BLASTN	pbcontigs.fasta	afu_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom1_nt	BLASTN	pbcontigs.fasta	afu_crom1_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom1_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom1_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom1_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom2_nt	BLASTN	pbcontigs.fasta	afu_crom2_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom2_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom2_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom2_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom3_nt	BLASTN	pbcontigs.fasta	afu_crom3_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom3_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom3_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom3_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom4_nt	BLASTN	pbcontigs.fasta	afu_crom4_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom4_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom4_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom4_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom5_nt	BLASTN	pbcontigs.fasta	afu_crom5_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom5_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom5_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom5_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom6_nt	BLASTN	pbcontigs.fasta	afu_crom6_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom6_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom6_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom6_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom7_nt	BLASTN	pbcontigs.fasta	afu_crom7_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom7_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom7_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom7_pbsinglets_tblastx.blast
	Aspergillus_fumigatus_crom8_nt	BLASTN	pbcontigs.fasta	afu_crom8_pbcontigs_blastn.blast
			pbsinglets.fasta	afu_crom8_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	afu_crom8_pbcontigs_tblastx.blast
			pbsinglets.fasta	afu_crom8_pbsinglets_tblastx.blast
Aspergillus nidulans	Aspergillus_nidulans_nt	BLASTN	pbcontigs.fasta	ani_pbcontigs_blastn.blast
			pbsinglets.fasta	ani_pbsinglets_blastn.blast
		TBLASTX	pbcontigs.fasta	ani_pbcontigs_tblastx.blast
			pbsinglets.fasta	ani_pbsinglets_tblastx.blast

Tabela 4.3: Comparação entre seqüências através do programa *blastall*.

liensis x cromossomo 1 do *Aspergillus fumigatus*, realizando comparações com BLASTN e TBLASTX.

A Figura 4.7 mostra a faixa de visualização de 5.000 bases da seqüência identificada como *gi_70996775*, que faz parte do cromossomo 1 do *Aspergillus fumigatus* e possui 2.671.808 pares de bases. O *gi - genInfo identifier* (identificador genInfo) é um identificador numérico de uma seqüência em bancos como o GenBank. Este identificador muda toda vez que ocorrem alterações nesta seqüência.

Para alcançarmos a visualização vista na Figura 4.7, inserimos a seguinte entrada no campo de busca da ferramenta: *Sequence:gi_70996775: 1..5000* ou

Organismo	**Arquivo BLAST** (blast_result_file)	**Seqüência** (reference_sequence_file) .fasta	**Saída** (gff_output_file)
Aspergillus fumigatus	afu_pbcontigs_blastn.blast	afu.fasta + pbcontigs.fasta	afu_pbcontigs_blastn.gff
	afu_pbsinglets_blastn.blast	afu.fasta + pbsinglets.fasta	afu_pbsinglets_blastn.gff
	afu_pbcontigs_tblastx.blast	afu.fasta + pbcontigs.fasta	afu_pbcontigs_tblastx.gff
	afu_pbsinglets_tblastx.blast	afu.fasta + pbsinglets.fasta	afu_pbsinglets_tblastx.gff
	afu_crom1_pbcontigs_blastn.blast	afu_crom1.fasta + pbcontigs.fasta	afu_crom1_pbcontigs_blastn.gff
	afu_crom1_pbsinglets_blastn.blast	afu_crom1.fasta + pbsinglets.fasta	afu_crom1_pbsinglets_blastn.gff
	afu_crom1_pbcontigs_tblastx.blast	afu_crom1.fasta + pbcontigs.fasta	afu_crom1_pbcontigs_tblastx.gff
	afu_crom1_pbsinglets_tblastx.blast	afu_crom1.fasta + pbsinglets.fasta	afu_crom1_pbsinglets_tblastx.gff
	afu_crom2_pbcontigs_blastn.blast	afu_crom2.fasta + pbcontigs.fasta	afu_crom2_pbcontigs_blastn.gff
	afu_crom2_pbsinglets_blastn.blast	afu_crom2.fasta + pbsinglets.fasta	afu_crom2_pbsinglets_blastn.gff
	afu_crom2_pbcontigs_tblastx.blast	afu_crom2.fasta + pbcontigs.fasta	afu_crom2_pbcontigs_tblastx.gff
	afu_crom2_pbsinglets_tblastx.blast	afu_crom2.fasta + pbsinglets.fasta	afu_crom2_pbsinglets_tblastx.gff
	afu_crom3_pbcontigs_blastn.blast	afu_crom3.fasta + pbcontigs.fasta	afu_crom3_pbcontigs_blastn.gff
	afu_crom3_pbsinglets_blastn.blast	afu_crom3.fasta + pbsinglets.fasta	afu_crom3_pbsinglets_blastn.gff
	afu_crom3_pbcontigs_tblastx.blast	afu_crom3.fasta + pbcontigs.fasta	afu_crom3_pbcontigs_tblastx.gff
	afu_crom3_pbsinglets_tblastx.blast	afu_crom3.fasta + pbsinglets.fasta	afu_crom3_pbsinglets_tblastx.gff
	afu_crom4_pbcontigs_blastn.blast	afu_crom4.fasta + pbcontigs.fasta	afu_crom4_pbcontigs_blastn.gff
	afu_crom4_pbsinglets_blastn.blast	afu_crom4.fasta + pbsinglets.fasta	afu_crom4_pbsinglets_blastn.gff
	afu_crom4_pbcontigs_tblastx.blast	afu_crom4.fasta + pbcontigs.fasta	afu_crom4_pbcontigs_tblastx.gff
	afu_crom4_pbsinglets_tblastx.blast	afu_crom4.fasta + pbsinglets.fasta	afu_crom4_pbsinglets_tblastx.gff
	afu_crom5_pbcontigs_blastn.blast	afu_crom5.fasta + pbcontigs.fasta	afu_crom5_pbcontigs_blastn.gff
	afu_crom5_pbsinglets_blastn.blast	afu_crom5.fasta + pbsinglets.fasta	afu_crom5_pbsinglets_blastn.gff
	afu_crom5_pbcontigs_tblastx.blast	afu_crom5.fasta + pbcontigs.fasta	afu_crom5_pbcontigs_tblastx.gff
	afu_crom5_pbsinglets_tblastx.blast	afu_crom5.fasta + pbsinglets.fasta	afu_crom5_pbsinglets_tblastx.gff
	afu_crom6_pbcontigs_blastn.blast	afu_crom6.fasta + pbcontigs.fasta	afu_crom6_pbcontigs_blastn.gff
	afu_crom6_pbsinglets_blastn.blast	afu_crom6.fasta + pbsinglets.fasta	afu_crom6_pbsinglets_blastn.gff
	afu_crom6_pbcontigs_tblastx.blast	afu_crom6.fasta + pbcontigs.fasta	afu_crom6_pbcontigs_tblastx.gff
	afu_crom6_pbsinglets_tblastx.blast	afu_crom6.fasta + pbsinglets.fasta	afu_crom6_pbsinglets_tblastx.gff
	afu_crom7_pbcontigs_blastn.blast	afu_crom7.fasta + pbcontigs.fasta	afu_crom7_pbcontigs_blastn.gff
	afu_crom7_pbsinglets_blastn.blast	afu_crom7.fasta + pbsinglets.fasta	afu_crom7_pbsinglets_blastn.gff
	afu_crom7_pbcontigs_tblastx.blast	afu_crom7.fasta + pbcontigs.fasta	afu_crom7_pbcontigs_tblastx.gff
	afu_crom7_pbsinglets_tblastx.blast	afu_crom7.fasta + pbsinglets.fasta	afu_crom7_pbsinglets_tblastx.gff
	afu_crom8_pbcontigs_blastn.blast	afu_crom8.fasta + pbcontigs.fasta	afu_crom8_pbcontigs_blastn.gff
	afu_crom8_pbsinglets_blastn.blast	afu_crom8.fasta + pbsinglets.fasta	afu_crom8_pbsinglets_blastn.gff
	afu_crom8_pbcontigs_tblastx.blast	afu_crom8.fasta + pbcontigs.fasta	afu_crom8_pbcontigs_tblastx.gff
	afu_crom8_pbsinglets_tblastx.blast	afu_crom8.fasta + pbsinglets.fasta	afu_crom8_pbsinglets_tblastx.gff
Aspergillus nidulans	ani_pbcontigs_blastn.blast	ani.fasta + pbcontigs.fasta	ani_pbcontigs_blastn.gff
	ani_pbsinglets_blastn.blast	ani.fasta + pbsinglets.fasta	ani_pbsinglets_blastn.gff
	ani_pbcontigs_tblastx.blast	ani.fasta + pbcontigs.fasta	ani_pbcontigs_tblastx.gff
	ani_pbsinglets_tblastx.blast	ani.fasta + pbsinglets.fasta	ani_pbsinglets_tblastx.gff

Tabela 4.4: Conversão dos arquivos de saída BLAST em arquivos no formato GFF através do programa *blast2gff*

simplesmente *gi_70996775: 1..5000*, pois a classe *Sequence* é automaticamente inserida pela ferramenta.

Em *Overview* temos a seqüência inteira e a região selecionada é mostrada em *Details*. O nível de ampliação é ajustado em *Scroll/Zoom*, onde neste caso é exibida uma região com 5.000 bases.

Na Figura 4.8 visualizamos as bases da seqüência mostrada na Figura 4.7, através de um clique na representação gráfica desta seqüência. O nome e o tamanho da seqüência são outras informações relevantes que podem ser vistas nesta figura.

Para conhecermos os alinhamentos da seqüência que representa o *Contig50* do *Paracoccidioides brasiliensis*, entramos com a seguinte informação no campo de busca de dados: *"Match: Contig50"*. O resultado pode ser visto na Figura 4.9. Nesta figura, temos alinhamentos deste *contig* com os dois segmentos do cromossomo 1 do *Aspergillus fumigatus*: *AFU1_95 id: gi_70996775 e AFU1_98, id: gi_70991345*. Podemos observar os escores obtidos e as regiões das seqüências

Organismo	Crom.	Arquivo GFF	Arquivo FASTA	Banco de Dados
Aspergillus fumigatus	Todos	afu_pbcontigs_blastn.gff + afu_pbcontigs_tblastx.gff	afu.fasta + pbcontigs.fasta	afu_pbcontigs_bdb
	1	afu_crom1_pbcontigs_blastn.gff + afu_crom1_pbcontigs_tblastx.gff	afu_crom1.fasta + pbcontigs.fasta	afu_crom1_pbcontigs_bdb
	2	afu_crom2_pbcontigs_blastn.gff + afu_crom2_pbcontigs_tblastx.gff	afu_crom2.fasta + pbcontigs.fasta	afu_crom2_pbcontigs_bdb
	3	afu_crom3_pbcontigs_blastn.gff + afu_crom3_pbcontigs_tblastx.gff	afu_crom3.fasta + pbcontigs.fasta	afu_crom3_pbcontigs_bdb
	4	afu_crom4_pbcontigs_blastn.gff + afu_crom4_pbcontigs_tblastx.gff	afu_crom4.fasta + pbcontigs.fasta	afu_crom4_pbcontigs_bdb
	5	afu_crom5_pbcontigs_blastn.gff + afu_crom5_pbcontigs_tblastx.gff	afu_crom5.fasta + pbcontigs.fasta	afu_crom5_pbcontigs_bdb
	6	afu_crom6_pbcontigs_blastn.gff + afu_crom6_pbcontigs_tblastx.gff	afu_crom6.fasta + pbcontigs.fasta	afu_crom6_pbcontigs_bdb
	7	afu_crom7_pbcontigs_blastn.gff + afu_crom7_pbcontigs_tblastx.gff	afu_crom7.fasta + pbcontigs.fasta	afu_crom7_pbcontigs_bdb
	8	afu_crom8_pbcontigs_blastn.gff + afu_crom8_pbcontigs_tblastx.gff	afu_crom8.fasta + pbcontigs.fasta	afu_crom8_pbcontigs_bdb
	Todos	afu_pbsinglets_blastn.gff + afu_pbsinglets_tblastx.gff	afu.fasta + pbsinglets.fasta	afu_pbsinglets.gff
	1	afu_crom1_pbsinglets_blastn.gff + afu_crom1_pbsinglets_tblastx.gff	afu_crom1.fasta + pbsinglets.fasta	afu_crom1_pbsinglets_bdb
	2	afu_crom2_pbsinglets_blastn.gff + afu_crom2_pbsinglets_tblastx.gff	afu_crom2.fasta + pbsinglets.fasta	afu_crom2_pbsinglets_bdb
	3	afu_crom3_pbsinglets_blastn.gff + afu_crom3_pbsinglets_tblastx.gff	afu_crom3.fasta + pbsinglets.fasta	afu_crom3_pbsinglets_bdb
	4	afu_crom4_pbsinglets_blastn.gff + afu_crom4_pbsinglets_tblastx.gff	afu_crom4.fasta + pbsinglets.fasta	afu_crom4_pbsinglets_bdb
	5	afu_crom5_pbsinglets_blastn.gff + afu_crom5_pbsinglets_tblastx.gff	afu_crom5.fasta + pbsinglets.fasta	afu_crom5_pbsinglets_bdb
	6	afu_crom6_pbsinglets_blastn.gff + afu_crom6_pbsinglets_tblastx.gff	afu_crom6.fasta + pbsinglets.fasta	afu_crom6_pbsinglets_bdb
	7	afu_crom7_pbsinglets_blastn.gff + afu_crom7_pbsinglets_tblastx.gff	afu_crom7.fasta + pbsinglets.fasta	afu_crom7_pbsinglets_bdb
	8	afu_crom8_pbsinglets_blastn.gff + afu_crom8_pbsinglets_tblastx.gff	afu_crom8.fasta + pbsinglets.fasta	afu_crom8_pbsinglets_bdb
Aspergillus nidulans	Todos	ani_pbcontigs_blastn.gff	ani.fasta + pbcontigs.fasta	ani_pbcontigs_blastn.gff
	Todos	ani_pbcontigs_tblastx.gff	ani.fasta + pbcontigs.fasta	ani_pbcontigs_tblastx_bdb
	Todos	ani_pbsinglets_blastn.gff	ani.fasta + pbsinglets.fasta	ani_pbsinglets_blastn_bdb
	Todos	ani_pbsinglets_tblastx.gff	ani.fasta + pbsinglets.fasta	ani_pbsinglets_tblastx_bdb

Tabela 4.5: Geração dos bancos de dados para o GBrowse a partir dos arquivos GFF e FASTA, através do programa *bp_load_gff.pl*

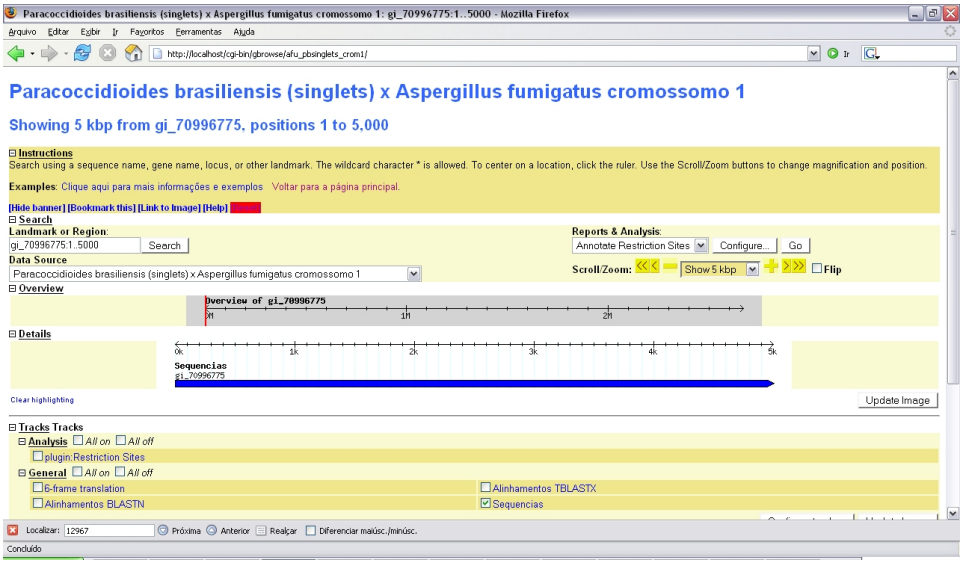


Figura 4.7: Visualização de parte da seqüência do cromossomo 1 do *Aspergillus fumigatus*.

```
description = Nome do Experimento
db_adaptor = Bio::DB::GFF
db_args = -adaptor berkeleydb -dsn caminho para o banco de dados
aggregators = match
reference class = Sequence
automatic classes = Sequence Match Hsp

[Sequences]
feature = sequence
glyph = generic
stranded = 1
bgcolor = blue
height = 10
key = Sequencias

[Alignments]
feature = match
glyph = segments
key = Alinhamentos
```

Tabela 4.6: Ajustes realizados em cada arquivo de configuração do GBrowse.

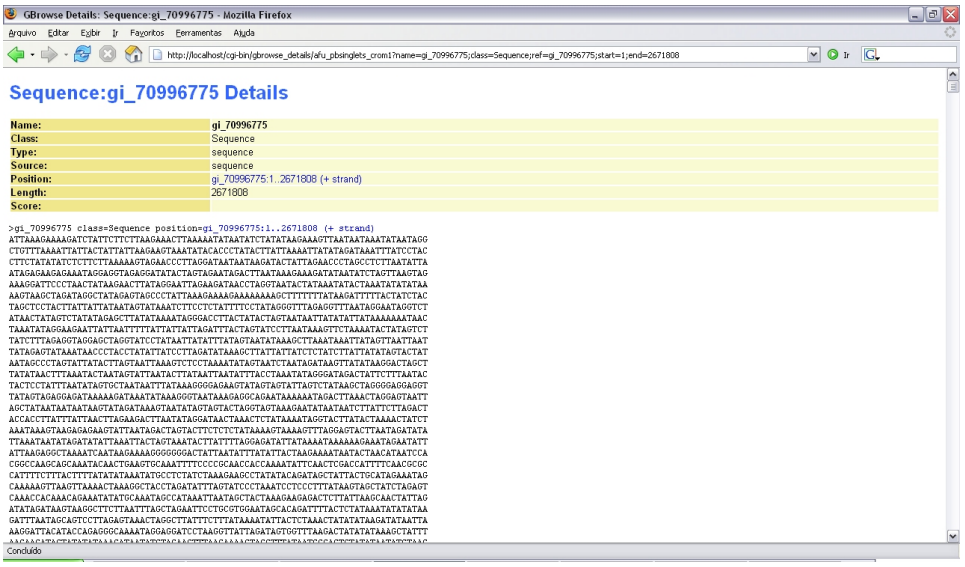


Figura 4.8: Visualização das bases que compõem a seqüência da Figura 4.7.

alvo de cada alinhamento.

Para visualizarmos os alinhamentos BLASTN do *Paracoccidioides brasiliensis* em relação às primeiras 5.000 bases do segmento *AFU1_95 (id: gi_70996775)* do cromossomo 1 do *Aspergillus fumigatus* (Figura 4.10), informamos no campo de busca de dados os seguintes valores: *"gi_70996775: 1..5000"*. Em seguida habilitamos a faixa de visualização "*Alinhamentos BLASTN*" e atualizamos a imagem

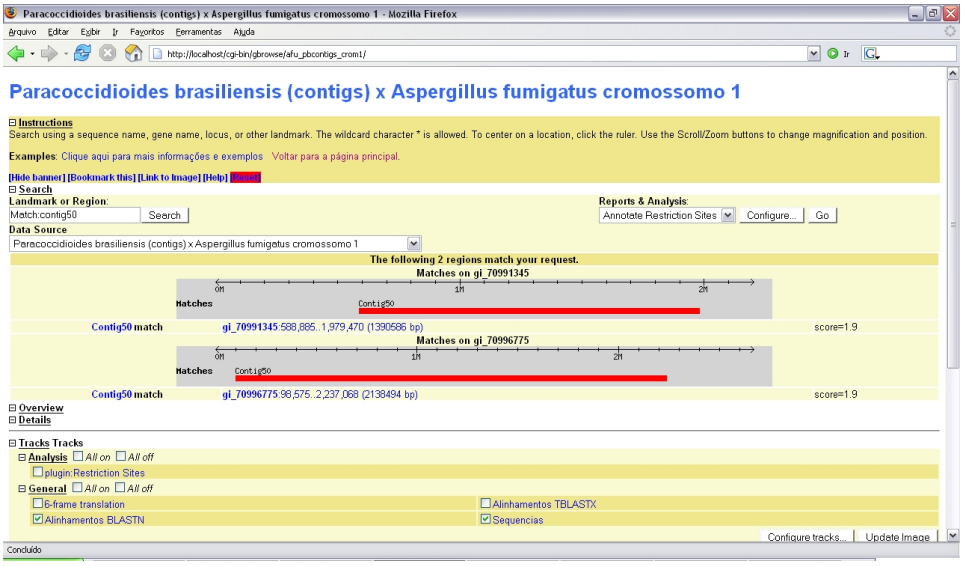


Figura 4.9: Visualização dos *matchs* da seqüência do *contig50* do *P. brasiliensis* com o cromossomo 1 do *A. fumigatus*.

(Update Image). Nesta figura, na parte *Details* temos a seqüência referência *(Aspergillus fumigatus)* e abaixo as regiões de cada *contig* onde houveram alinhamentos com esta seqüência.

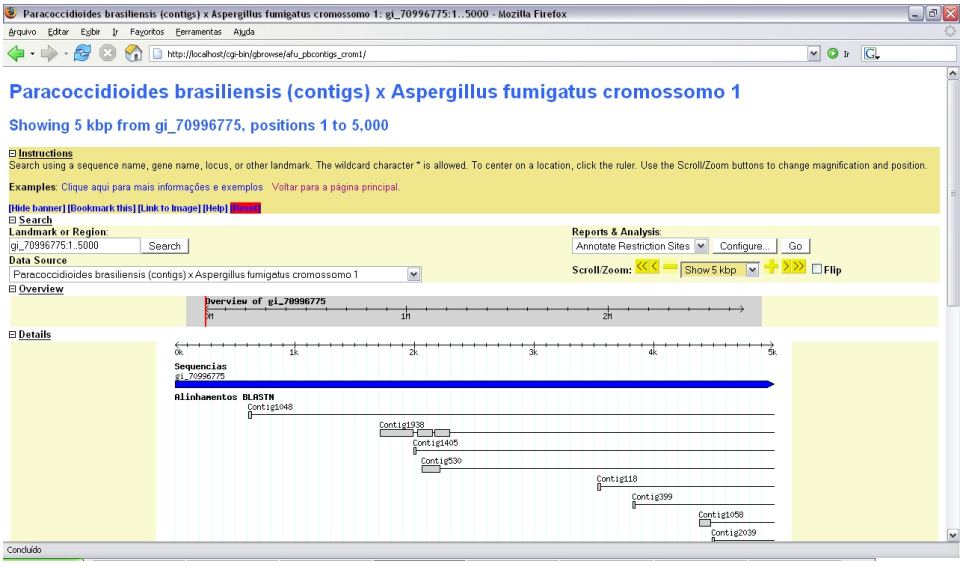


Figura 4.10: Visualização dos alinhamentos BLASTN do *P. brasiliensis* com as primeiras 5.000 bases da seqüência de uma das partes do cromossomo 1 do *A. fumigatus*.

O mesmo procedimento pode ser feito para a visualização dos alinhamentos TBLASTX. Neste caso, é preciso habilitar adicionalmente a faixa de visualização *"Alinhamentos TBLASTX"*. A Figura 4.11 mostra os alinhamentos BLASTN e

TBLASTX sendo exibidos simultaneamente em relação à seqüência do cromossomo 1 do *Aspergillus fumigatus*.

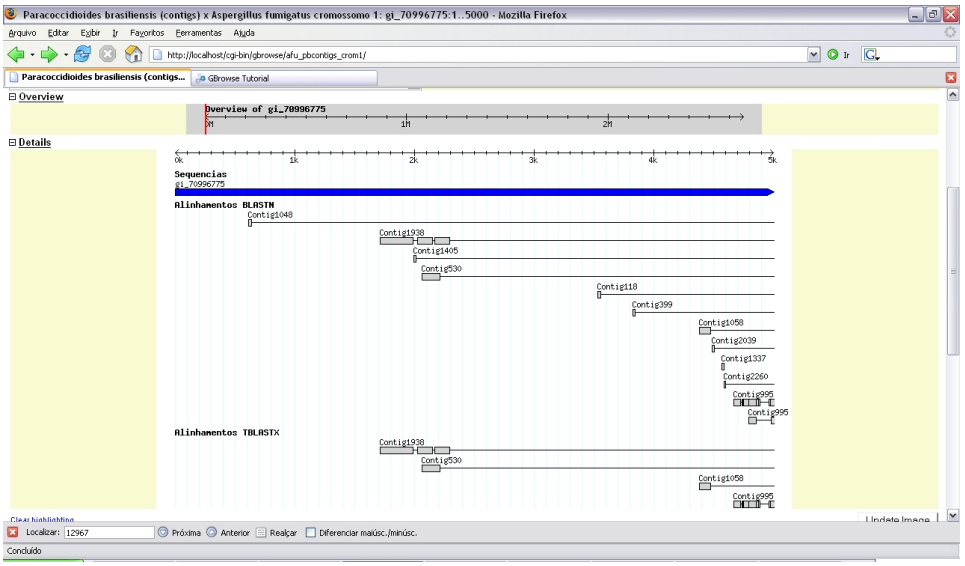


Figura 4.11: Visualização dos alinhamentos BLASTN e TBLASTX do *P. brasiliensis* com as primeiras 5.000 bases da seqüência de uma parte do cromossomo 1 do *A. fumigatus*.

Para visualizarmos as seis possíveis molduras de leitura de uma região da seqüência do cromossomo 1 do *Aspergillus fumigatus*, selecionamos a faixa de visualização *6-frame translation*. Na Figura 4.12 exibimos uma região da seqüência de 5.000 bases, notando que os aminoácidos que compõem as molduras de leitura não podem ser lidos pois o nível de detalhe (*Zoom*) está muito baixo. O intuito deste exemplo é mostrar o uso da função *Scroll/Zoom*, que será mostrada na próxima figura.

Para visualização legível das bases das molduras de leitura, ajustamos a ampliação de 5.000 bases, vista na figura anterior, para apenas 200 bases. Neste caso, podemos observar claramente as bases que formam as molduras (Figura 4.13). O recurso de ampliação pode ser usado em todas as faixas de visualização disponíveis.

Neste capítulo, apresentamos alguns exemplos do que a ferramenta GBrowse pode oferecer com base nos dados disponíveis em nossos Projetos Genoma. À medida que mais informações forem coletadas, poderemos incrementar estes bancos de dados com novas faixas de visualização como Motifs e codificação de proteínas entre outras, tornando-os cada vez mais completos.

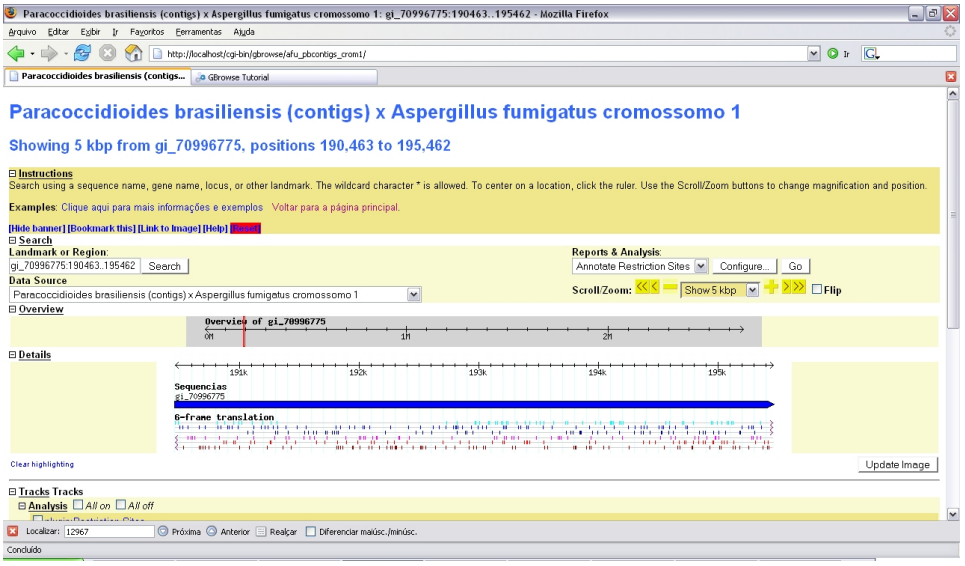


Figura 4.12: Visualização das seis molduras de leitura de uma região contendo 5.000 bases da seqüência.

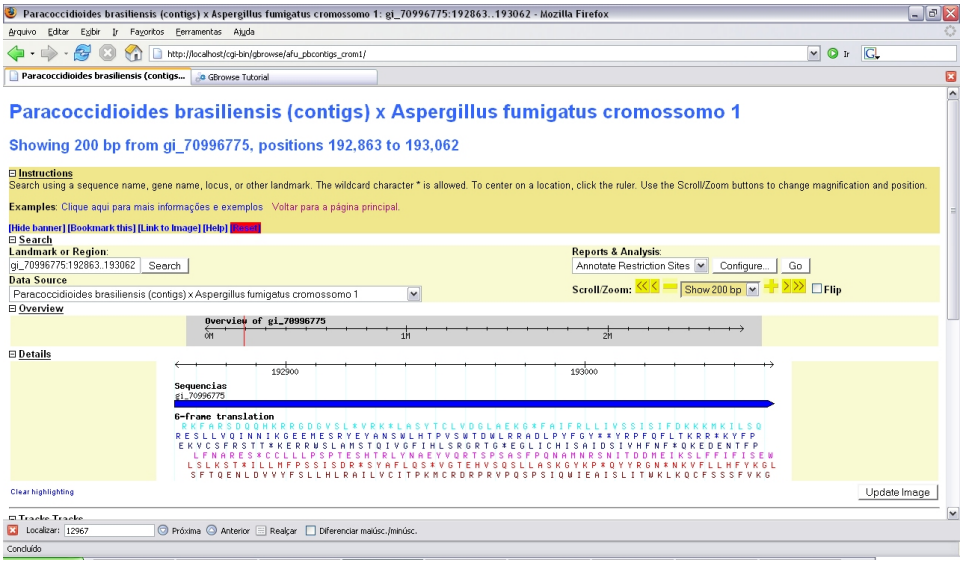


Figura 4.13: Visualização das seis molduras de leitura de uma região contendo 200 bases da seqüência.

Contribuições para a Ferramenta

Durante o preparo dos experimentos, visto na seção anterior, detectamos algumas dificuldades na utilização da ferramenta GBrowse.

A principal dificuldade relatada foi quanto a não familiaridade com a sintaxe utilizada para a localização de informações. Como vimos na Seção 3.5, a ferramenta possui uma sintaxe própria para a busca de dados, que varia de acordo com o tipo de informação desejada.

Com o objetivo de facilitar o uso da ferramenta, principalmente no período

inicial de adaptação com sua sintaxe, foram feitas algumas contribuições na *interface* com o usuário, permitindo uma melhor compreensão da organização dos bancos de dados inseridos na ferramenta e sua rápida utilização sem exigir prévios conhecimentos da sintaxe utilizada. As contribuições realizadas foram:

- Geração de uma página inicial com *links* para todos os bancos de dados incluídos no GBrowse (Figura 4.14).
  Esta página torna mais fácil a compreensão da organização dos bancos de dados além de permitir o acesso direto a cada um deles.

- Geração de um *link* de informações e exemplos.
  Através deste *link*, uma nova página para cada banco de dados foi desenvolvida (Figura 4.15), contendo uma breve descrição do banco de dados, informando os organismos envolvidos, os tipos de comparações realizadas e uma série de *links* para os principais tipos de informações de cada banco.

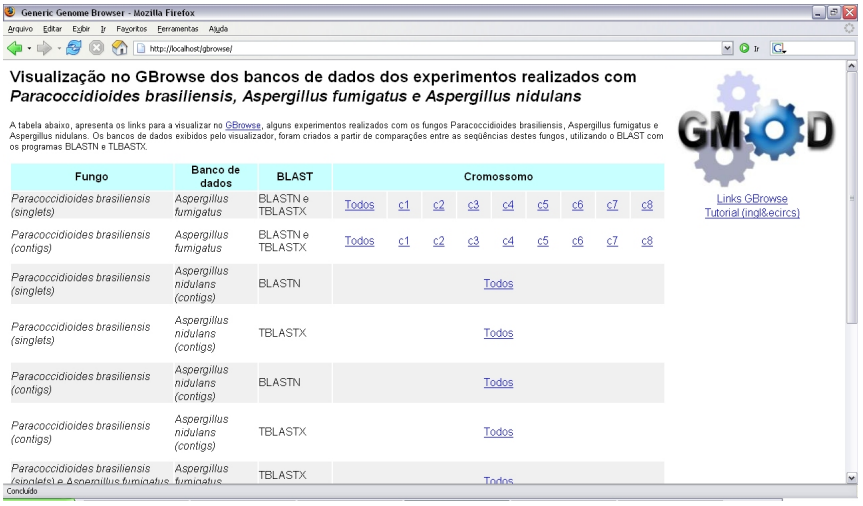


Figura 4.14: Página inicial com *links* para todos os bancos de dados incluídos no GBrowse.

As visualizações das comparações entre os ESTs do *P. brasiliensis* e os DNAs genômicos do *A. fumigatus* e *A. nidulans* permitirão inferir a organização[1] dos genes do *P. brasiliensis* dentro do seus cromossomos. A partir do modelo *A. fumigatus*, sintenias[2] entre genes poderão ser identificadas. A partir destas inferências, experimentos biológicos poderão ser elaborados para confirmação de sintenias entre genes do *P. brasiliensis*.

[1]Localização física relativa de genes nos cromossomos.
[2]Localização física de genes dentro do mesmo cromossomo.

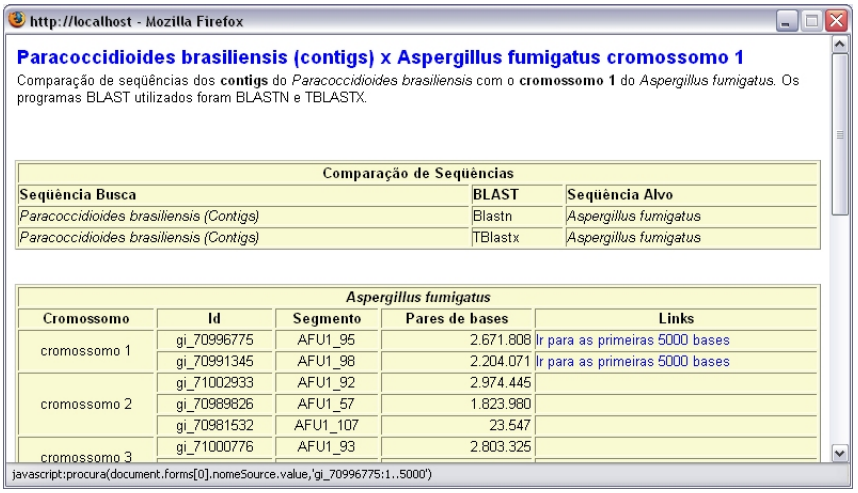


Figura 4.15: Página de informações e exemplos para cada banco de dados dentro do GBrowse.

Capítulo 5

Ferramenta para Visualização de Enzimas em Mapas de Vias Metabólicas

Os bancos de dados de vias do KEGG contêm os mapas de vias metabólicas representados em XML e podem ser visualizados graficamente através da *Internet*. Para esta visualização, o pesquisador precisa entrar manualmente com os códigos de cada enzima e informar o mapa de vias desejado. Este processo leva tempo e está sujeito a falhas na digitação dos dados. Em alguns casos, a localização das enzimas é feita de forma totalmente manual.

Para automatizar este processo, possibilitando ainda a oferta de outros serviços, construímos uma ferramenta para visualização de genes em mapas de vias metabólicas do KEGG. Esta ferramenta possui uma arquitetura *Web* e multi-plataforma e utiliza a API *Web Services* e os bancos de dados de vias em XML (KGML) oferecidos pelo KEGG.

Portanto, o objetivo deste capítulo é descrever a construção desta ferramenta. Na Seção 5.1 faremos a descrição da especificação da ferramenta, definindo o escopo, abordando os modelos de uso e os requisitos funcionais e não funcionais. Em seguida, na Seção 5.2, descreveremos o projeto para a construção da ferramenta, mostrando a infra-estrutura necessária, detalhes da arquitetura, modelagem estática e detalhamento dos casos de uso. Por último, na Seção 5.3, descreveremos experimentos para mostrar o uso da ferramenta, fazendo uma breve discussão.

5.1 Especificação

Nesta seção faremos a especificação da ferramenta proposta, chamada *PathwayView*. Esta especificação foi feita segundo informações levantadas junto aos

usuários do Laboratório de Biologia Molecular da UnB.

5.1.1 Definição do Escopo

A ferramenta tem como missão apoiar pesquisas que utilizam vias metabólicas do KEGG, fornecendo um mecanismo de localização de enzimas nestes mapas e permitindo a visualização dos mesmos.

5.1.2 Modelos de Uso

O modelo de uso é um modelo dinâmico que descreve como o sistema interage com seu ambiente [53]. Para representar este modelo, utilizamos o diagrama de casos de uso da Linguagem de Modelagem Unificada - UML [47]. Os casos de uso da ferramenta *PathwayView* podem ser vistos na Figura 5.1.

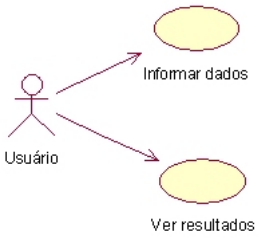


Figura 5.1: Os dois casos de uso da ferramenta *PathwayView*.

Caso de uso: Informar Dados
O usuário deve informar os seguintes dados à ferramenta: as enzimas a serem localizadas nos mapas de vias metabólicas, o tipo de resultado desejado e os mapas de vias metabólicas desejados.

Caso de uso: Ver resultados
A ferramenta deve apresentar uma lista de resultados contendo o nome de cada via metabólica e os dados das respectivas enzimas localizadas. Através desta lista, o usuário poderá visualizar cada mapa desejado. O usuário pode solicitar esta lista como um arquivo.

5.1.3 Requisitos Funcionais

Os requisitos funcionais podem ser definidos como as funções que um sistema deve possuir. Os requisitos funcionais da ferramenta *PathwayView* serão descritos a seguir.

A ferramenta deve permitir que as enzimas de entrada sejam informadas através de um arquivo ou pela digitação direta no formulário de dados. Em ambos os casos, um formato de entrada apropriado deve ser definido, contando com informações mínimas como o *Enzyme Commission Number - EC*, anotação (nome da enzima), grupo de cada enzima e opcionalmente um *link* HTML.

A ferramenta deve ainda disponibilizar uma lista estática de mapas de vias metabólicas para que o usuário possa selecionar em quais desses mapas as enzimas de entrada serão procuradas. Caso o usuário não utilize esta lista estática, a ferramenta deve procurar as enzimas de entrada em todos os mapas (lista dinâmica) mais recentes armazenados no banco de dados remoto do KEGG.

Quando a seleção de mapas de vias metabólicas ocorrer pela lista estática, o sistema poderá fornecer a opção de mostrar no resultado somente aqueles mapas onde foram localizados pelo menos uma enzima de entrada. Quando a seleção de mapas de vias metabólicas ocorrer pela lista dinâmica, a ferramenta poderá fornecer a opção de mostrar somente os mapas onde todas as enzimas de entrada forem localizadas.

A ferramenta deve gerar o resultado na forma de uma lista, onde serão mostrados os nomes dos mapas de vias metabólicas e as enzimas localizadas. Cada mapa com alguma enzima localizada poderá ser visualizado através desta lista. Um relatório de saída contendo a lista deve estar disponível para ser recuperada.

5.1.4 Requisitos Não Funcionais

Os requisitos não funcionais dizem respeito aos requisitos de desempenho, qualidade e restrições no projeto de um sistema. Para esta ferramenta, os requisitos não funcionais definem a tecnologia utilizada e a portabilidade esperada: a ferramenta deve ser implementada utilizando tecnologia *open source* ou software livre e deve ser portável para diferentes plataformas.

5.2 Projeto

Nesta seção descrevemos o projeto da ferramenta, baseado na especificação levantada na seção anterior.

O projeto e o desenvolvimento utilizarão a abordagem orientada a objetos. Nesta abordagem, as operações e funções da ferramenta são descritas em termos de objetos e interações entre eles [53].

5.2.1 Infra-estrutura

A infra-estrutura da ferramenta é composta por (Figura 5.2):

- Servidor *Web*: servidor de páginas *Web J2EE* onde residirá a ferramenta
- *Web Service* KEGG: *Web Service* responsável por fornecer os serviços para a ferramenta através da *Internet*
- Navegador *Web*: aplicativo de navegação de páginas *Web*, utilizado pelo usuário para acesso à ferramenta.

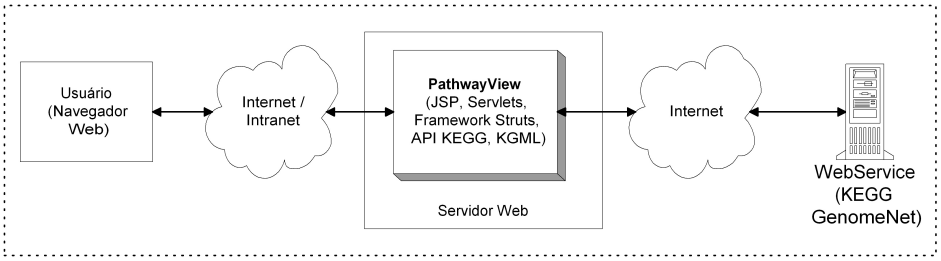


Figura 5.2: Visão geral da infra-estrutura da ferramenta *PathwayView*.

5.2.2 Arquitetura

A arquitetura da ferramenta (Figura 5.3) é composta pelos seguintes componentes tecnológicos:

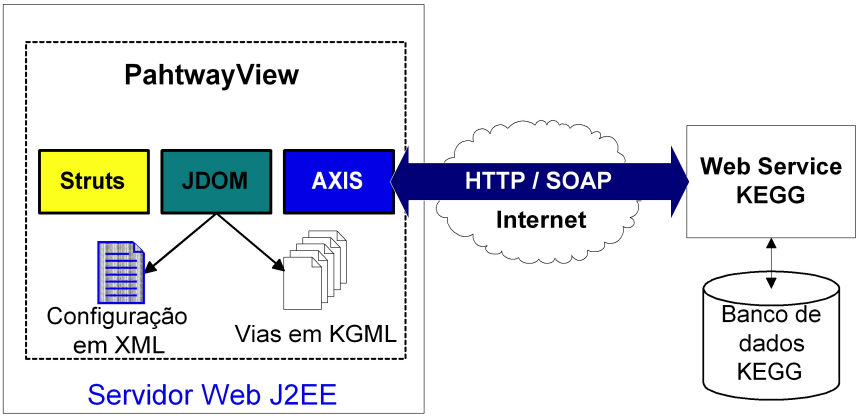


Figura 5.3: Arquitetura macro da ferramenta PathwayView.

Java/J2EE: a ferramenta será uma aplicação *Web* programada em *Java/J2EE*. *Java* é uma linguagem de programação orientada a objetos e J2EE é um conjunto

de diferentes APIs que estendem as funcionalidades da linguagem. As principais APIs J2EE utilizadas serão as que suportam o desenvolvimento para *Internet*, como *Servlets* e *JSP*.

Servlets é uma tecnologia Java que estende a funcionalidade de um servidor Web, provendo mecanismos que permitem uma aplicação Web invocar componentes (Classes) de negócios localizados no servidor. Estes componentes podem acessar toda à família de APIs Java e os demais recursos disponíveis no lado do servidor.

JSP (JavaServer Pages) é uma tecnologia Java que provê recursos para a criação de interfaces Web com o usuário com conteúdo dinâmico. Esta tecnologia utiliza instruções (tags) no estilo XML para encapsular a lógica que gera o conteúdo da página e HTML mais XML para a formatação da interface.

Framework Struts: este framework possui código aberto e é bastante utilizada no desenvolvimento de aplicações para *Internet* (aplicações *Web*).

O *framework Struts* é baseado no paradigma Modelo-Visão-Controle - MVC. Neste paradigma, o Modelo representa as regras do sistema, a Visão representa a interface com o usuário e o Controle é a gerência do fluxo da informação na aplicação. O modelo MVC desacopla a interface com o usuário da lógica de negócio e dados. A Figura 5.4 apresenta uma visão geral do framework Struts.

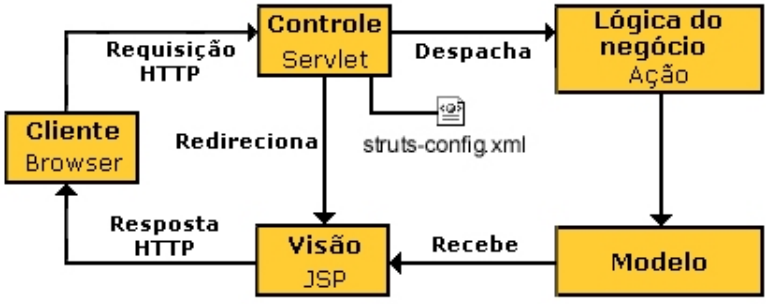


Figura 5.4: Visão geral do framework Struts.

- **Modelo:** o framework Struts não provê um modelo de componentes especializados para lidar com a camada de modelo, deixando a cargo do desenvolvedor utilizar o modelo que mais lhe convier para representar sua lógica de negócio.

- **Controle:** O componente de controle do Struts são a espinha dorsal de uma aplicação Web. Este componente é implementado utilizando Servlets

chamado *ActionServlet*. Este servlet é responsável por receber todas as requisições dos clientes da aplicação Web e delegar o controle de cada uma destas requisições para um classe (classe de Ação) determinada pelo desenvolvedor. Uma vez que a classe de Ação completa seu processamento, ela retorna o controle para a classe *ActionServlet* que irá determinar qual a Visão irá apresentar o resultado.

- **Visão:** cada componente de visão do framework Struts é mapeado para uma página escrita utilizando a tecnologia JSP. Esta página normalmente contém além de HTML, tags especiais do Struts para auxiliar na geração de conteúdo dinâmico.

API KEGG: A API é um *Web Service* fornecido pelo KEGG que possui diversos serviços como consultas e geração de imagens de mapas de vias metabólicas extraídas do banco de dados de vias KEGG.

Dentre outros, os resultados retornados por estes serviços são listas de dados e *links* HTML que ficam temporariamente armazenados nos servidores KEGG. Os serviços utilizados foram:

- **mark_pathway_by_objects (pathway_id, object_id_list):** marca os objetos informados no parâmetro object_id_list no mapa de vias informado no parâmetro pathway_id e retorna a URL da imagem gerada. A seguir mostramos um exemplo da chamada deste serviço, onde o mapa de vias desejado é identificado por path:map00472 e as enzimas são ec:1.1.1.1 e ec:1.1.1.2: mark_pathway_by_objects ("path:map00472", ["ec:1.1.1.1","ec:1.1.1.2"]);

- **color_pathway_by_objects (pathway_id, object_id_list, fg_color_list, bg_color_list):** colore no mapa de vias informado no parâmetro pathway_id com os objetos informados no parâmetro object_id_list. As cores do texto / borda e fundo de cada objeto são respectivamente informadas nos parâmetros fg_color_list e bg_color_list. Este serviço retorna a URL para a imagem gerada. A seguir mostramos um exemplo da chamada deste serviço, onde o mapa de vias desejado é identificado por path:map00472, as enzimas são ec:1.1.1.1 e ec:1.1.1.2 e as cores de texto e fundo para cada enzima são representadas pelo conjunto ["#ff0000", "#00ff00"]: mark_pathway_by_objects ("path:map00472", ["ec:1.1.1.1","ec:1.1.1.2"], ["#ff0000", "#00ff00"], ["#ff0000", "#00ff00"]). Neste exemplo utilizamos as mesmas cores de texto e fundo para os dois objetos informados;

- **get_html_of_marked_pathway_by_objects (pathway_id, object_id_list):** versão do método mark_pathway_by_objects que retorna um HTML no lugar da imagem. A HTML retornada possui embutida uma imagem do mapa com os objetos marcados. Vários objetos contidos neste HTML possuem links que remetem para páginas do KEGG com informações mais detalhadas sobre eles;

- **get_html_of_colored_pathway_by_objects (pathway_id, object_id_list, fg_color_list, bg_color_list):** versão do método color_pathway_by_object que retorna um HTML no lugar da imagem. A HTML retornada possui embutida uma imagem do mapa com seus objetos coloridos. Assim como ocorre com o serviço anterior, links apontam para informações mais detalhadas de vários objetos do mapa;

- **get_pathways_by_enzymes (enzyme_id_list):** retorna as vias que possuem todas as enzimas informadas no parâmetro enzyme_id_list. A seguir mostramos um exemplo da chamada deste serviço, onde o mapa de vias desejado é identificado por path:map00472: get_pathways_by_enzymes ("path: map00472").

Axis: Apache Axis [9] é uma implementação de código livre do protocolo SOAP. O Apache Axis encapsula os detalhes como codificação de mensagens enviadas a um *Web Service*, a decodificação da resposta no cliente, RPC (Remote Procedure Call - Chamada a Procedimentos Remotos), SOAP e XML entre outros, tornando mais simples e rápido o desenvolvimento de clientes de *Web Services* e dos próprios *Web Services*.

Uma importante ferramenta disponível no pacote Apache Axis e utilizada na construção da ferramenta PathwayView foi a WSDL2Java. A WSDL2Java gera, a partir do WSDL de um *Web Service*, as classes clientes (stubs) e as classes servidoras (skeletons) necessárias para acessar este *Web Service* através do Java.

JDOM: JDOM [31] é uma API para facilitar a manipulação de XML em Java. Esta API oferece uma solução completa para acessar, manipular e escrever dados em XML dentro de um código Java.

A linguagem Java implementa duas APIs para acesso e manipulação de XML: a API DOM (Document Object Model for XML), que é a implementação do padrão estipulado pela W3C *(World Wide Web Consortium)* [59] e a API SAX *(Simple API for XML)* que é baseada em eventos.

JDOM é uma API de código livre que pode ser utilizada como alternativa a ambas APIs citadas anteriormente. Assim como DOM, JDOM representa um documento XML como uma árvore composta de elementos, atributos, comentários e textos entre outros. Esta árvore é carregada na memória de onde qualquer parte pode ser acessada e modificada.

KGML: todos os mapas de vias metabólicas armazenados localmente estão representando em XML através da KGML. Estes mapas são utilizados na montagem da lista hierárquica estática com os nomes dos mapas e para eventual procura das enzimas de entrada.

5.2.3 Modelagem Estática: Diagramas de Classes

O modelo estático descreve a estrutura estática do sistema em termos das classes de objetos de sistema e de seus relacionamentos [53].

Esta seção mostra as funcionalidades do sistema através da modelagem estática, utilizando o diagrama de classes da UML.

A ferramenta possui dois pacotes de classes: *pathway*, que contém as classes de negócio (Figura 5.5) e *action*, que possui as classes de controle (Figura 5.6).

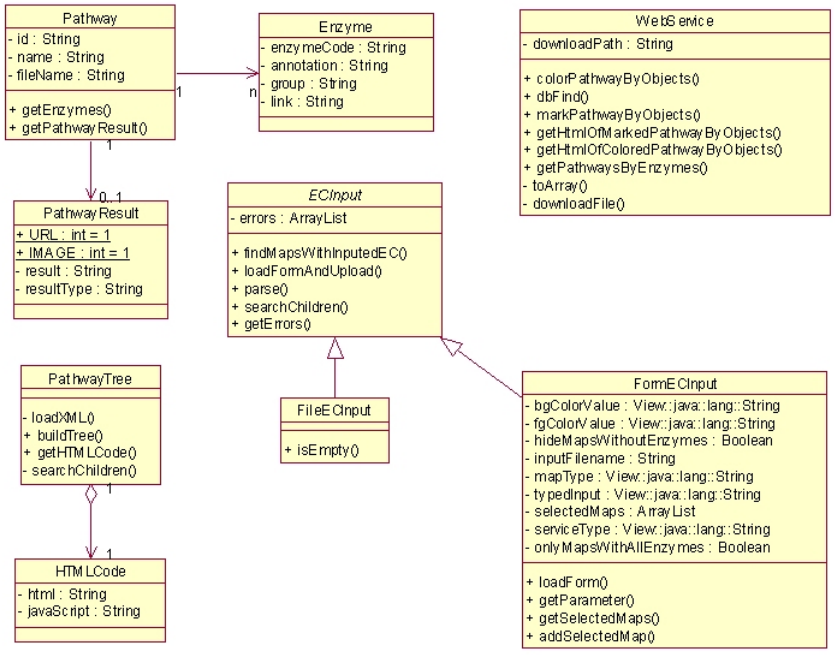


Figura 5.5: Diagrama de classes do pacote *pathway*.

As classes do pacote de negócio *pathway* são descritas na Tabela 5.1, enquanto as classes do pacote de controle *action* são descritas na Tabela 5.2.

Classe	Descrição
Pathway	Representa as informações básicas de um mapa vias metabólicas.
Enzyme	Representa as informações básicas de uma enzima.
PathwayResult	Representa o resultado obtido quando um determinado mapa de vias metabólicas é enviado ao KEGG para execução.
Web Service	Contém os serviços disponíveis no *Web Service* do KEGG.
ECInput	Representação abstrata das enzimas de entrada.
FormECInput	Serviços oferecidos quando as enzimas de entrada são informadas via formulário.
FileECInput	Serviços oferecidos quando as enzimas de entrada são informadas via arquivo.
PathwayTree	Fornece os métodos necessários para a construção da árvore hierárquica estática de mapas de vias metabólicas que podem ser selecionados.
HTMLCode	Parte agregada da classe PathwayTree que representa o código HTML e *JavaScript* gerados na construção da árvore hierárquica estática de mapas de vias metabólicas.

Tabela 5.1: Descrição das classes do pacote de negócio *pathway*.

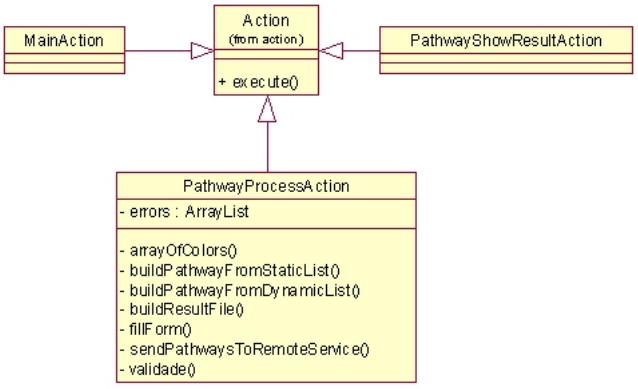


Figura 5.6: Diagrama de classes do pacote *action*.

Classe	Descrição
Action	Classe da framework struts que contém o método de controle executado em cada fluxo do sistema.
MainAction	Classe generalizada de controle cuja ação é construir a página de entrada de dados do sistema.
PathwayShowResultAction	Classe generalizada de controle cuja ação é mostrar a página de resultados do sistema.
PathwayProcessAction	Classe generalizada de controle cuja ação é realizar o processamento principal, como o upload do arquivo de entrada, procura dos mapas de vias metabólicas que contêm as entradas e o acionamento do *Web Service* entre outros.

Tabela 5.2: Descrição das classes do pacote de controle *action*.

5.2.4 Detalhamento dos casos de uso

Nesta parte, descreveremos detalhes relevantes ao projeto dos dois casos de uso da ferramenta.

Formato de Dados para as Enzimas de Entrada

As enzimas de entrada que se deseja localizar nos mapas de vias metabólicas podem ser informadas através de um arquivo texto ou através da digitação dos dados no formulário de entrada. Em ambos os casos, as informações devem estar de acordo com o formato mostrado na Figura 5.7.

```
# arquivo de entrada da ferramenta PathwayView
# ec;anotação;grupo;link
3.1.3.48;Protein-tyrosine-phosphatase;Contig118;
2.7.2.3;Thioredoxin;Contig1929;http://localhost/gbrowse/projetoPb/showDetails?name=Contig1929
5.4.2.1;Phosphoglycerate mutase;PBDRV-Y1-104t_D08;
1.1.2.4;PBGRJ-M1-422t_G08;D-lactate ferricytochrome c oxidoreductase;
```

Figura 5.7: Formato do arquivo com as enzimas de entrada.

Neste formato, cada linha representando uma enzima de entrada possui as seguintes colunas:

- **ec:** Enzyme Commission Number, que é o código da enzima que se deseja marcar nos mapas de vias. Este código segue o padrão estabelecido pelo KEGG e deve ser informado.
- **anotação:** anotação da enzima, que deve ser informada.
- **grupo:** nome do grupo do gene. Normalmente é a identificação do gene dentro de um projeto genoma e também deve ser informado.
- **link:** um *link* no formato URL[1] para uma página externa ao sistema. Esta opção permite estabelecer por exemplo uma ligação com os dados disponíveis no GBrowse.

Cada coluna tem seu valor separado pelo caractere ponto e vírgula, com exceção da última coluna, onde este caractere pode ser omitido. O caractere # é utilizado para comentar uma linha.

Escolha dos mapas de vias metabólicas

Quanto a escolha dos mapas de vias metabólicas pode-se optar por uma das duas opções mostradas a seguir.

A primeira opção, chamada de *seleção estática*, permite escolher um ou mais mapas de vias metabólicas de um conjunto atual de 129 mapas. Neste caso, podemos informar que no resultado devem estar somente os mapas de vias metabólicas que contêm alguma enzima de entrada localizada.

[1]Universal Resource Locator é o endereço único de um arquivo na *Internet*.

A vantagem desta opção é que podemos filtrar a escolha dos mapas de vias metabólicas, selecionando somente aqueles de interesse. A desvantagem é que o conjunto de mapas pode tornar-se desatualizado com o tempo, caso o desenvolvedor não atualize a lista de mapas.

A segunda opção, chamada de *descoberta dinâmica*, permite utilizar a lista de mapas de vias metabólicas mais recente, descobrindo em tempo real aqueles mapas que possuam alguma enzima de entrada. Neste caso, podemos informar que no resultado devem estar somente os mapas que contêm todas as enzimas de entrada localizadas.

A vantagem desta opção é que o sistema sempre utilizará os mapas de vias metabólicas mais recentes, sem a necessidade de atualização do sistema. A desvantagem é que não poderemos escolher em quais mapas a localização deve ser feita.

A lista estática de mapas de vias metabólicas está contida em um arquivo XML (Figura 5.8), que possui os nomes dos mapas de vias metabólicas organizados hierarquicamente e pode ser modificado para incorporar novos mapas, remover mapas existentes e alterar a hierarquia dos mapas.

```
<?xml version="1.0" encoding="UTF-8"?> <pathway-maps>
  <maps title="Metabolic Pathways">
    <maps title="Amino Acid Metabolism">
       <map id="map00251" title="Glutamate metabolism"
            file="map00251.xml"/>
       <map id="map00252" title="Alanine and aspartate metabolism"
            file="map00252.xml"/>
       <map id="map00271" title="Methionine metabolism"
            file="map00271.xml"
    </maps>
    <maps title="Nucleotide Metabolism">
       <map id="map00230" title="Purine metabolism"
            file="map00230.xml" />
       <map id="map00240" title="Pyrimidine metabolism"
            file="map00240.xml" />
    </maps>
  </maps>
</pathway-maps>
```

Figura 5.8: Arquivo XML contendo a lista estática de mapas de vias metabólicas.

Tipos de resultado

A ferramenta oferece quatro tipos de resultados diferentes:

1. Gerar Imagens de Mapas de Vias com enzimas coloridas: neste tipo, cada mapa de vias metabólicas listado no resultado é fornecido como um arquivo de imagem. As enzimas de entrada localizadas em cada mapa têm sua borda e fundo coloridos conforme informado;

2. Gerar Imagens de Mapas de Vias com enzimas marcadas: mesmo caso anterior, exceto pela forma de identificação das enzimas no mapa, que neste caso têm somente a borda colorida conforme informado;

3. Gerar HTML de Mapas de Vias com enzimas coloridas: neste tipo, cada mapa de vias metabólicas listado no resultado é fornecido como uma página HTML e cada enzima de entrada localizada no mapa tem sua borda e fundo coloridos conforme informado;

4. Gerar HTML de Mapas de Vias com enzimas marcadas: mesmo caso anterior, exceto pela forma de identificação das enzimas no mapa, que neste caso têm somente a borda colorida conforme informado;

Para os resultados dos tipos 3 e 4, a página HTML retornada possui *links* para diversas informações contidas no mapa de vias metabólicas, como informações detalhadas de uma determinada enzima.

Escolha de cores

É possível que o usuário determine quais serão as cores utilizadas nos retângulos que representam as enzimas localizadas nos mapas de vias metabólicas. Para isto, a ferramenta contará com um formulário de escolha das cores da borda e preenchimento destes retângulos.

Relatório de saída

O resultado do processamento pode ser obtido através do relatório de saída. Este relatório será disponibilizado ao final de cada processamento. O relatório possui o formato mostrado na Figura 5.9.

5.2.5 Interface

A interface principal da ferramenta (Figura 5.10) é a responsável pela entrada de dados e possuirá as seguintes informações: nome do arquivo com as enzimas de entrada, campo para digitação das enzimas de entrada, seleção de um entre quatro tipos de resultados, seleção das cores de fundo e borda, seleção dos mapas de vias ou seleção dinâmica.

==========
PathwayView report gerado em ¡data e hora¿
==========

Arquivo de entrada:
Nro de enzimas de entrada:
Tipo de resultado:
Seleção de mapas:
Ocultar mapas sem enzimas de entrada: *disponível somente para seleção estática de mapas*
Somente mapas com todas as enzimas de entrada: *disponível somente para seleção dinâmica de mapas*
==========

1. Enzimas de entrada
==========
lista de enzimas no formato: EC - anotação - grupo - link
2. Vias metabólicas pesquisadas
==========
lista de mapas de vias metabólicas no formato: nome da via
cada mapa de vias metabólicas possui uma outra lista com as enzimas localizadas: EC - anotação - grupo

Figura 5.9: Formato do relatório de saída da ferramenta *PathwayView*.

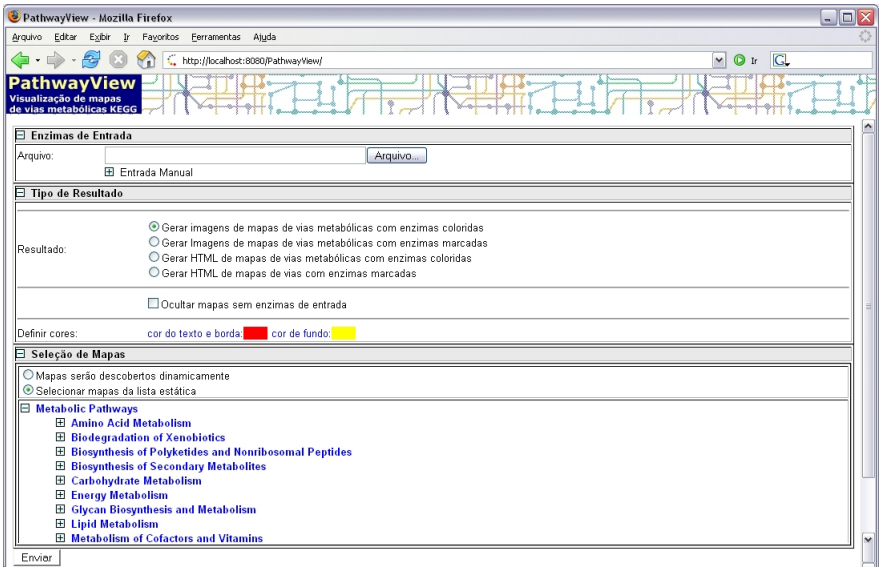


Figura 5.10: Interface de entrada de dados da ferramenta *PathwayView*.

5.3 Experimentos e Discussão

Nesta seção demonstraremos a utilização da ferramenta *PathwayView* através da realização de diversos experimentos. Os dados utilizados são resultantes de experimentos de macroarranjo de cDNA, gerados pelo Laboratório de Biologia Molecular da Universidade de Brasília [5] e possuem informações de genes diferencialmente expressos do fungo *Paracoccidioides brasiliensis* nas formas de micélio (M) e de levedura (Y).

Os dados foram enviados ao Laboratório de Bioinformática em dois arquivos (um para cada forma do *Paracoccidioides brasiliensis*). O arquivo com o *Paracoccidioides brasiliensis* na forma micélio continha 58 genes e na forma de levedura continha 270 genes. Destes genes foram selecionados somente aqueles que codificam alguma enzima, pois são os que podem ser localizadas nos mapas de vias

metabólicas do KEGG através do código da enzima *Enzyme Commission Number - EC*. Ao término da seleção, obtivemos 18 genes na forma de micélio e 121 genes na forma de levedura.

A partir destas informações, construímos um arquivo de entrada com as enzimas na forma de micélio e um para na forma de levedura, nomeados respectivamente de *pb_m_ec.txt* e *pb_y_ec.txt*. A montagem destes arquivos obedeceu o formato de entrada da ferramenta *PathwayView*. A Figura 5.11 mostra o arquivo de entrada *pb_m_ec.txt*.

```
# arquivo pb_m_ec.txt
# ec;anotação;grupo;link
Contig8; 1.10.2.2;ubiquinol-cytochrome c reductase iron-sulpher subunit precursor
Contig203; 3.2.1.58;1,3-beta-glucosidase
Contig552; 6.3.2.19;ubiquitin-conjugating enzyme
Contig1929; 1.8.1.9;Thioredoxin
PBDCR-M1-011t_C06; 2.7.7.4;Ras GTPase superfamily
PBDEX-M1-006t_A08; 3.1.21.-;ATPase, NSFA, protein involved in protein transport between ...
PBDEX-M1-026t_G04;6.3.5.5;carbamyl phosphate synthetase
PBDEX-Y1-020t_A12;3.1.3.16;protein phosphatase 2c homolog 1
PBDEX-Y1-023t_A02;2.7.1.-;SRPK1-like Kinase in Yeast
PBDEX-Y1-026t_G06;1.1.1.41;isocitrate dehydrogenase
PBDEX-Y1-032t_B06; 3.6.1.3;26S protease subunit and member of the CDC48/PAS1/SEC18 ...
PBDEX-Y1-032t_H01; 2.7.7.7;DNA polymerase delta large chain
PBDEX-Y1-035t_A07; 2.7.1.48;Uridine kinase
PBDEX-Y1-037t_A06; 5.99.1.2;topoisomerase I
PBDMO-Y1-009t_G02; 2.5.1.54;3-deoxy-D-arabino-heptulosonate 7-phosphate (DAHP) synthase ...
PBDRV-Y1-041t_E01; 2.7.1.2;GLUCOKINASE (GLUCOSE KINASE)
PBGEX-Y1-071t_G10; 2.7.1.48;similarity to uridine kinase
PBGEX-Y1-088t_G12; 6.2.1.5;beta subunit of succinyl-CoA ligase
```

Figura 5.11: Arquivo de entrada contendo as enzimas do *P. brasiliensis* na forma de micélio.

A Tabela 5.3 mostra o resumo dos experimentos realizados com estes arquivos, contendo o número do experimento, o tipo do arquivo utilizado (Micélio ou Levedura) no experimento, o número de enzimas de entrada em cada arquivo, o tipo de seleção de mapas (Estática ou Dinâmica), se foi aplicado algum ajuste (filtro), o número de mapas por número de enzimas de entrada localizadas (0, 1, 2, 3 e >3), a enzima mais vezes localizada e por último o mapa com maior número de enzimas diferentes localizadas. As duas últimas colunas exibem entre parênteses as respectivas quantidades obtidas.

Experimento 1: tomando inicialmente o arquivo *pb_m_ec.txt*, informamos sua localização à ferramenta, selecionamos a opção "Gerar imagens de mapas de vias metabólicas com enzimas coloridas" e escolhemos todos os mapas de vias metabólicas da lista estática disponível. O objetivo deste experimento foi localizar as enzimas diferencialmente expressas na forma de micélio em todos os

Experimento	Arquivo	Enzimas	Estática/Dinâmica	Ajuste	Mapas com enzimas	Número de enzimas localizadas					Enzima mais localizada	Mapa com maior número de enzimas localizadas
						0	1	2	3	>3		
1	M	18	E	**N**	18	111	14	3	1	0	Glucokinase (4) Beta subunit of succinyl-CoA ligase (4)	Pyrimidine metabolism (3)
2	M	18	E	**S**	18	0	14	3	1	0	Glucokinase (4) Beta subunit of succinyl-CoA ligase (4)	Pyrimidine metabolism (3)
3	M	18	**D**	**N**	45	0	32	10	2	1	SRPK1-like Kinase in Yeast (21)	Pyrimidine metabolism (5)
4	M	18	**D**	**S**	0	0	0	0	0	0	-	-
5	Y	121	E	**N**	78	51	29	17	12	20	glycosyltransferase (10)	Glycolysis / Gluconeogenesis (10)
6	Y	121	E	**S**	78	0	29	17	12	20	glycosyltransferase (10)	Glycolysis / Gluconeogenesis (10)
7	Y	121	**D**	**N**	118	0	31	32	22		glycosyltransferase (26)	Glycolysis / Gluconeogenesis (15)
8	Y	121	**D**	**S**	0	0	0	0	0	0	-	-

Tabela 5.3: Experimentos realizados com a ferramenta *PahtwayView* utilizando os dados de macroarranjo de cDNA do fungo *Paracoccidioides brasiliensis*.

mapas de vias metabólicas disponíveis na ferramenta. A Figura 5.12 mostra o resultado do processamento.

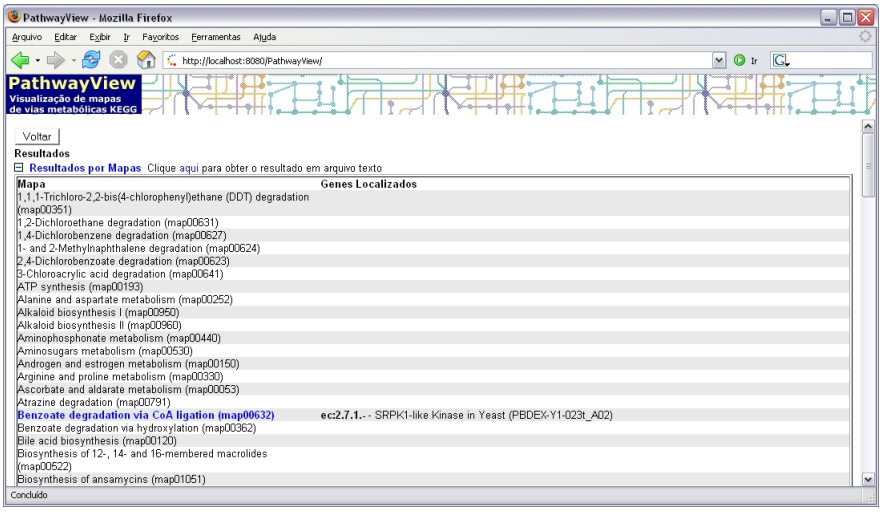


Figura 5.12: Resultado do processamento do arquivo *pb_m_ec.txt*, selecionando todas as vias metabólicas da lista estática.

Obtivemos 18 mapas de vias metabólicas com pelo menos uma das enzimas de entrada, onde 14 mapas com 1 enzima localizada, 3 mapas com 2 enzimas

localizadas e 1 mapa com 3 enzimas localizadas (Tabela 5.3). Nesta tabela notamos que todos os mapas, inclusive aqueles que não possuem nenhuma destas enzimas, estão presentes no resultado. As enzimas mais localizadas foram *Glucokinase* e *Beta subunit of succinyl-CoA ligase*, ambas em 4 mapas cada. O mapa *Pyrimidine metabolism* possui o maior número de enzimas diferentes localizadas.

Experimento 2: para eliminarmos os mapas de vias metabólicas que não possuem enzimas localizadas, selecionamos a opção "Ocultar mapas sem enzimas de entrada"na tela de entrada de dados da ferramenta. A Figura 5.13 apresenta a tela de saída obtida após a eliminação. Os resultados são quase idênticos ao experimento anterior, com exceção que neste caso não são apresentados os mapas que não possuem enzimas localizadas (Tabela 5.3).

Mapa	Genes Localizados
Benzoate degradation via CoA ligation (map00632)	ec:2.7.1.- - SRPK1-like Kinase in Yeast (PBDEX-Y1-023t_A02)
C5-Branched dibasic acid metabolism (map00660)	ec:6.2.1.5 - beta subunit of succinyl-CoA ligase (PBGEX-Y1-088t_G12)
Citrate cycle (TCA cycle) (map00020)	ec:6.2.1.5 - beta subunit of succinyl-CoA ligase (PBGEX-Y1-088t_G12) ec:1.1.1.41 - isocitrate dehydrogenase (PBDEX-Y1-026t_G06)
Galactose metabolism (map00052)	ec:2.7.1.2 - GLUCOKINASE (GLUCOSE KINASE) (PBDRV-Y1-041t_E01)
Glutamate metabolism (map00251)	ec:6.3.5.5 - carbamyl phosphate synthetase (PBDEX-M1-026t_G04)
Glycolysis / Gluconeogenesis (map00010)	ec:2.7.1.2 - GLUCOKINASE (GLUCOSE KINASE) (PBDRV-Y1-041t_E01)
Inositol phosphate metabolism (map00562)	ec:2.7.1.- - SRPK1-like Kinase in Yeast (PBDEX-Y1-023t_A02)
Nicotinate and nicotinamide metabolism (map00760)	ec:2.7.1.- - SRPK1-like Kinase in Yeast (PBDEX-Y1-023t_A02)
Oxidative phosphorylation (map00190)	ec:1.10.2.2 - ubiquinol-cytochrome c reductase iron-sulpher subunit precursor (Contig8)
Phenylalanine, tyrosine and tryptophan biosynthesis (map00400)	ec:2.5.1.54 - 3-deoxy-D-arabino-heptulosonate 7-phosphate (DAHP) synthase isoenzyme (PBDMO-Y1-009t_G02)
Propanoate metabolism (map00640)	ec:6.2.1.5 - beta subunit of succinyl-CoA ligase (PBGEX-Y1-088t_G12)
Purine metabolism (map00230)	ec:3.6.1.3 - 26S protease subunit and member of the CDC48/PAS1/SEC18 family of ATPases Rpt3p [Saccharomyces cerevisiae] (PBDEX-Y1-032t_B06) ec:2.7.7.4 - Ras GTPase superfamily (PBDCR-M1-011t_C06)
Pyrimidine metabolism (map00240)	ec:6.3.5.5 - carbamyl phosphate synthetase (PBDEX-M1-026t_G04) ec:2.7.1.48 - Uridine kinase (PBDEX-Y1-035t_A07) ec:1.8.1.9 - Thioredoxin (Contig1929)
Reductive carboxylate cycle (CO2 fixation) (map00720)	ec:6.2.1.5 - beta subunit of succinyl-CoA ligase (PBGEX-Y1-088t_G12)
Selenoamino acid metabolism (map00450)	ec:2.7.7.4 - Ras GTPase superfamily (PBDCR-M1-011t_C06)
Starch and sucrose metabolism (map00500)	ec:2.7.1.2 - GLUCOKINASE (GLUCOSE KINASE) (PBDRV-Y1-041t_E01) ec:3.2.1.58 - 1,3-beta-glucosidase (Contig203)
Streptomycin biosynthesis (map00521)	ec:2.7.1.2 - GLUCOKINASE (GLUCOSE KINASE) (PBDRV-Y1-041t_E01)
Sulfur metabolism (map00920)	ec:2.7.7.4 - Ras GTPase superfamily (PBDCR-M1-011t_C06)

Figura 5.13: Resultado do processamento do arquivo *pb_m_ec.txt*, selecionando todas as vias metabólicas da lista estática e filtrando somente as vias com enzimas localizadas.

Nesta figura, temos à esquerda os mapas de vias metabólicas e à direita as enzimas localizadas nos respectivos mapas. Note que alguns mapas como "Purine metabolism (map00230)" e "Pyrimidine metabolism (map00240)" possuem mais de uma enzima localizada.

Pode-se abrir a imagem gerada do mapa de vias metabólicas "Purine metabolism (map00230)" através de um clique sobre o *link* disponível. A imagem do mapa de vias metabólicas é mostrada na Figura 5.14.

Note que nesta figura, as enzimas localizadas através do código da enzima estão representados de acordo com as cores informadas na tela de entrada de dados.

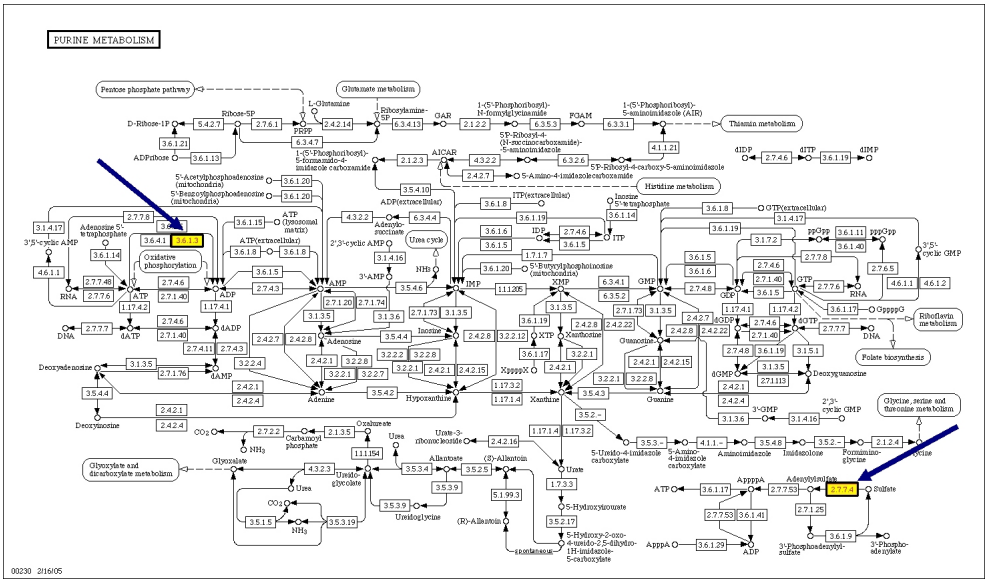


Figura 5.14: Imagem do mapa de vias metabólicas "Purine metabolism" com enzimas localizadas (em destaque).

Voltando à tela de resultados, podemos obter o relatório de saída com os resultados gerados, através do *link* "Clique aqui para obter o resultado em arquivo texto". A Figura 5.15 exibe trecho deste arquivo.

Experimento 3: neste experimento exploramos o recurso de descoberta dinâmica de mapas, onde a lista mais recente de mapas de vias metabólicas é obtida em tempo real.

Neste caso, como mostra a Tabela 5.3, temos um expressivo aumento no número de mapas com enzimas localizadas em relação aos experimentos anteriores. Este aumento deve-se justamente ao fato da utilização do método de descoberta dinâmica de mapas, onde a ferramenta utiliza todos os tipos de mapas (mapas de vias metabólicas e mapas de vias regulatórias entre outros) mais recentes encontrados no banco de dados de vias remoto do KEGG.

Experimento 4: neste experimento aplicamos adicionalmente ao experimento anterior o ajuste que informa à ferramenta para localizar somente os mapas com todas as enzimas de entrada localizadas. Como mostra a Tabela 5.3, nenhum mapa de vias preencheu este requisito.

Experimentos 5 a 8: os experimentos 5, 6, 7 e 8 repetem respectivamente os experimentos 1, 2, 3 e 4, alterando o arquivo de entrada de micélio (*pb_m_ec.txt*) para o arquivo de levedura (*pb_m_ec.txt*). Os dados destes experimentos também

```
============================================================
PathwayView report gerado em 08/02/2006 14:05:43
============================================================

Arquivo de entrada: pb_m_ec.txt
Nro de enzimas de entrada: 18
Tipo de resultado: Gerar imagens de mapas de vias metabólicas com enzimas coloridas
Seleção de mapas: Selecionar mapas da lista estática
Ocultar mapas sem enzimas de entrada: Sim
============================================================

1. Enzimas de entrada
============================================================
ec:1.10.2.2 - ubiquinol-cytochrome c reductase iron-sulpher subunit precursor - Contig8 -
ec:3.2.1.58 - 1,3-beta-glucosidase - Contig203 -
ec:6.3.2.19 - ubiquitin-conjugating enzyme - Contig552 -
ec:1.8.1.9 - Thioredoxin - Contig1929 -
ec:2.7.7.4 - Ras GTPase superfamily - PBDCR-M1-011t_C06 -
ec:3.1.21.- - ATPase, NSFA, protein involved in protein transport between endoplasmic reticulum and Golg -
PBDEX-M1-006t_A08 -
ec:6.3.5.5 - carbamyl phosphate synthetase - PBDEX-M1-026t_G04 -
ec:3.1.3.16 - protein phosphatase 2c homolog 1 - PBDEX-Y1-020t_A12 -
ec:2.7.1.- - SRPK1-like Kinase in Yeast - PBDEX-Y1-023t_A02 -
ec:1.1.1.41 - isocitrate dehydrogenase - PBDEX-Y1-026t_G06 -
ec:3.6.1.3 - 26S protease subunit and member of the CDC48/PAS1/SEC18 family of ATPases Rpt3p [Saccharomyces
cerevisiae] - PBDEX-Y1-032t_B06 -
ec:2.7.7.7 - DNA polymerase delta large chain - PBDEX-Y1-032t_H01 -
ec:2.7.1.48 - Uridine kinase - PBDEX-Y1-035t_A07 -
ec:5.99.1.2 - topoisomerase I - PBDEX-Y1-037t_A06 -
ec:2.5.1.54 - 3-deoxy-D-arabino-heptulosonate 7-phosphate (DAHP) synthase isoenzyme - PBDMO-Y1-009t_G02 -
ec:2.7.1.2 - GLUCOKINASE (GLUCOSE KINASE) - PBDRV-Y1-041t_E01 -
ec:2.7.1.48 - similarity to uridine kinase - PBGEX-Y1-071t_G10 -
ec:6.2.1.5 - beta subunit of succinyl-CoA ligase - PBGEX-Y1-088t_G12 -

2. Vias metabólicas pesquisadas
============================================================
Benzoate degradation via CoA ligation
≫ ec:2.7.1.- - SRPK1-like Kinase in Yeast(PBDEX-Y1-023t_A02)
------------------------------
C5-Branched dibasic acid metabolism
≫ ec:6.2.1.5 - beta subunit of succinyl-CoA ligase(PBGEX-Y1-088t_G12)
------------------------------
Citrate cycle (TCA cycle)
≫ ec:6.2.1.5 - beta subunit of succinyl-CoA ligase(PBGEX-Y1-088t_G12)
≫ ec:1.1.1.41 - isocitrate dehydrogenase(PBDEX-Y1-026t_G06)
------------------------------
```

Figura 5.15: Parte do relatório de saída da ferramenta *PathwayView*.

podem ser vistos na Tabela 5.3.

Esta ferramenta contribui para processar um grande volume de dados, facilitando a localização das vias metabólicas e das enzimas diferenciais dentro destas vias. No caso específico do *P. brasiliensis*, este estudo permitirá inferir como ocorre a adaptação biológica deste patógeno nos seres humanos, o que poderá indicar novas possibilidades para o desenvolvimento de drogas ou fungicidas.

Capítulo 6

Conclusões e Trabalhos Futuros

Neste trabalho, inicialmente estudamos diversas ferramentas para visualização de dados genômicos comparativos. Dentre elas, escolhemos a ferramenta GBrowse que descrevemos com maiores detalhes. Utilizando o GBrowse, desenvolvemos um método para facilitar a visualização de dados comparativos, que forneceu um modelo para criação de bancos de dados contendo as informações a serem visualizadas. Este método visa facilitar, agilizar e documentar o processo para criação de novos bancos de dados no GBrowse, que não é trivial. Para demonstrarmos a utilização do método, realizamos experimentos com as ESTs do fungo do *Paracoccidioides brasiliensis* e os DNAs genômicos dos fungos *Aspergillus fumigatus* e *Aspergillus nidulans*. As visualizações geradas por estes experimentos serão utilizadas para inferir organização de genes do *Paracoccidioides brasiliensis* dentro dos seus cromossomos.

Além disso, desenvolvemos um programa, chamado *PathwayView*, para localizar enzimas nos mapas de vias metabólicas do KEGG e para a visualizar estes mapas. Com esta ferramenta, realizamos experimentos utilizando genes diferencialmente expressos de *Paracoccidioides brasiliensis*, identificados por experimentos de macroarranjo de cDNA no Laboratório de Biologia Molecular da UnB. A *PathwayView* contribui para processar um grande volume de dados, facilitando a localização das vias metabólicas e das enzimas diferenciais dentro destas vias. Em *Paracoccidioides brasiliensis*, este estudo permitirá aprofundar o conhecimento de como ocorre a adaptação biológica deste patógeno no ser humano, o que abre novas perspectivas para o desenvolvimento de drogas ou fungicidas.

O método apresentado para a preparação de dados genômicos visualizados na ferramenta GBrowse deve ser realizado em sucessivas etapas com execuções de programas e montagem de arquivos. Este processo poderia ser automatizado através do desenvolvimento de uma ferramenta computacional que utilizasse parâmetros, onde todas as informações necessárias seriam previamente fornecidas

e ao final seriam gerados os bancos de dados no padrão do GBrowse. Isto contribuiria para acelerar o longo processo de geração de novos bancos de dados. Este método também poderia ser expandido, incluindo no banco de dados de saída outros tipos de informações como codificação de proteínas e *motifs*.

A ferramenta para localização e visualização de enzimas em mapas de vias metabólicas poderia ser aprimorada através da implementação de diversas funções disponíveis na API KEGG, como obtenção de genes através do organismo ou enzima e obtenção de enzimas através de reação ou vice-versa. Outra funcionalidade interessante seria a elaboração de uma estrutura para armazenamento e recuperação das imagens representando as vias metabólicas obtidas como resultado, evitando processamentos repetidos de informações. Adicionalmente, a criação de relatórios estatísticos facilitaria a análise das informações e a implementação de uma versão utilizando *Java* e o pacote gráfico *Swing* dispensaria o uso de um servidor *web*.

Este trabalho visa contribuir para construção de ferramentas para visualização de dados genômicos, que é uma área importante de pesquisa em Bioinformática, devido ao enorme volume de dados textuais disponíveis nos bancos e arquivos gerados por projetos genoma em todo o mundo.

Apêndice A

Serviços para vias fornecidas pela API KEGG

Método	Descrição	Retorno
Colorindo vias		
mark_pathway_by_objects (pathway_id, object_id_list)	Marca os objetos no mapa de vias e retorna a URL da imagem gerada. Exemplos de object_id: 'eco:b4258', 'cpd:C00135', 'ko:K01881'.	string (URL)
color_pathway_by_objects (pathway_id, object_id_list, fg_color_list, bg_color_list)	Colore os objetos correspondentes ao 'object_id_list' no mapa de vias com as cores especificadas e retorna a URL para a imagem colorida. 'fg_color_list' é usado para especificar a cor do texto e borda do objeto e 'bg_color_list' é usado para o cor de fundo.	string (URL)
color_pathway_by_elements (pathway_id, element_id_list, fg_color_list, bg_color_list)	Colore os objetos correspondentes ao 'element_id_list' no mapa de vias com as cores especificadas e retorna a URL para a imagem colorida. 'fg_color_list' é usado para especificar a cor do texto e borda do objeto e 'bg_color_list' é usado para o cor de fundo. O 'element_id' é um identificador numérico único no banco de dados de vias. Exemplos de element_id: 78, 79, 41, 47.	string (URL)
get_html_of_marked_pathway_by_objects (pathway_id, object_id_list)	Versão do método 'mark_pathway_by_objects' que retorna um HTML ao invés da imagem. A HTML possui a imagem do mapa com links para vários elementos deste mapa.	string (URL)
get_html_of_colored_pathway_by_objects (pathway_id, object_id_list, fg_color_list, bg_color_list)	Versão do método 'color_pathway_by_object' que retorna um HTML ao invés da imagem. A HTML possui a imagem do mapa com links para vários elementos deste mapa.	string (URL)
get_html_of_colored_pathway_by_elements (pathway_id, element_id_list, fg_color_list, bg_color_list)	Versão do método 'color_pathway_by_elements' que retorna um HTML ao invés da imagem. A HTML possui a imagem do mapa com links para vários elementos deste mapa.	string (URL)
Objetos nas vias		
get_elements_by_pathway (pathway_id)	Retorna todos os objetos da via especificada.	ArrayOfPathwayElement
get_genes_by_pathway (pathway_id)	Retorna todos os genes da via especificada	ArrayOfstring (genes_id)
get_enzymes_by_pathway (pathway_id)	Retorna todas as enzimas da via especificada	ArrayOfstring (enzyme_id)
get_compounds_by_pathway (pathway_id)	Retorna todas as combinações da via especificada.	ArrayOfstring (compound_id)
get_glycans_by_pathway (pathway_id)	Retorna todos os glycans da via especificada.	ArrayOfstring (glycan_id)
get_reactions_by_pathway (pathway_id)	Retorna todas as reações da via especificada.	ArrayOfstring (reaction_id)
get_kos_by_pathway (pathway_id)	Retorna todos os KOs da via especificada.	ArrayOfstring (ko_id)
Vias por objetos		
get_pathways_by_genes (genes_id_list)	Retorna todas as vias que incluem todos os genes informados.	ArrayOfstring (pathway_id)
get_pathways_by_enzymes (enzyme_id_list)	Retorna todas as vias que incluem todos as enzimas informadas.	ArrayOfstring (pathway_id)
get_pathways_by_compounds (compound_id_list)	Retorna todas as vias que incluem todas as combinações informadas.	ArrayOfstring (pathway_id)
get_pathways_by_glycans (glycan_id_list)	Retorna todas as vias que incluem todos os glycans informados.	ArrayOfstring (pathway_id)
get_pathways_by_reactions (reaction_id_list)	Retorna todas as vias que incluem todos as reações informadas.	ArrayOfstring (pathway_id)
get_pathways_by_kos (ko_id_list, org)	Retorna todas as vias que incluem todos os KOs informados.	ArrayOfstring (pathway_id)
Relação entre vias		
get_linked_pathways (pathway_id)	Retorna todas as vias que são ligadas à via informada.	ArrayOfstring (pathway_id)

Tabela A.1: Lista de funcionalidades para vias disponíveis na API KEGG [32].

Apêndice B

Arquivo de configuração do GBrowse

```
[GENERAL]
description = Paracoccidioides brasiliensis (\emph{contigs}) x
Aspergillus fumigatus
db_adaptor  = Bio::DB::GFF
db_args     = -adaptor berkeleydb
              -dsn '/gbrowse/databases/afu_pbcontigs_bdb'

aggregators = match
              processed_transcript

plugins = Aligner RestrictionAnnotator

# lista de faixas selecionadas por padrão
default features = Sequences

reference class  = Sequence

# exemplos mostrados na introdução
examples = contig100 Match:contig50

# classes que serão testadas automaticamente quando a mesma não for
informada na pesquisa
automatic classes = Sequence Match Hsp

### HTML a ser inserido em determinadas partes na página ###
head =
header =
footer =
html1 =
html2 =
html3 =
html4 =
html5 =
html6 =

# tamanhos (em pixels) das imagens exibidas
```

```
image widths  = 450 640 800 1024

# tamanho (em pixels) padrão das imagens exibidas
default width = 800

# configuração de estilos e diretórios do site Web
stylesheet  = /gbrowse/gbrowse.css
buttons     = /gbrowse/images/buttons
tmpimages   = /gbrowse/tmp

# tamanho (em pares de bases) máximo e padrão do tamanho do segmento
a ser exibido
max segment     = 50000
default segment = 5000

# níveis de ampliação (em pares de bases)
zoom levels    = 100 200 1000 2000 5000 10000 20000 40000 50000

# cores usadas nas faixas visão geral e detalhada
overview bgcolor = lightgrey
detailed bgcolor = lightgoldenrodyellow


############################
# configuração de um Plugin
############################
[Aligner:plugin]
alignable_tracks   = EST
upcase_tracks      = CDS Motifs
upcase_default     = CDS

################################
# ajustes padrões para as faixas
################################
[TRACK DEFAULTS]
glyph         = generic
height        = 10
bgcolor       = lightgrey
fgcolor       = black
font2color    = blue
label density = 25
bump density  = 100

# link HTML usado ao clicar na visão detalhada
link          = AUTO


## CONFIGURAÇÕES DAS FAIXAS #########
```

```
# configurações das faixas do projeto
#####################################

[Sequences]
feature      = sequence
glyph        = generic
stranded     = 1
bgcolor      = blue
height       = 10
key          = Sequencias

[AlignmentsBlastn]
feature      = match
glyph        = segments
key          = Alinhamentos BLASTN

[AlignmentsBlastn:30000]
glyph        = box

[AlignmentsBlastn:45000]
glyph        = box
bump         = 0

[AlignmentsTBlastx]
feature      = match:tblastx
glyph        = segments
key          = Alinhamentos TBLASTX

[AlignmentsTBlastx:30000]
glyph        = box

[AlignmentsTBlastx:45000]
glyph        = box
bump         = 0

[Translation]
glyph          = translation
global feature = 1
height         = 40
fgcolor        = purple
start_codons   = 0
stop_codons    = 1
translation    = 6frame
key            = 6-frame translation
```

Apêndice C

Glossário de termos da Biologia Molecular

- **Anticódon** é uma seqüência complementar reversa do códon.
- **Códon** é um conjunto formado por cada três bases de uma seqüência de mRNA.
- **Cromossomo** é uma molécula de DNA, que tem como função armazenar as características hereditárias de uma espécie.
- **DNA** é o ácido desoxiribonucleico, formado por duas cadeias de moléculas que possuem uma estrutura helicoidal, descoberta por Watson e Crick em 1953 [60].
- **ESTs** - Expressed Sequence Tags (ou etiquetas de seqüências expressas) são seqüências curtas de fragmentos de mRNA.
- **Gene** é um segmento do DNA que contém as informações genéticas para a codificação de proteínas.
- **Genoma** de um organismo é o seu conjunto completo de cromossomos.
- **Molduras de leitura** *(reading frame)* são seqüências de *códons* formadas a partir de um *códon de início*.
- **Proteína** é uma cadeia de moléculas de aminoácidos.
- **RNA** é o ácido ribonucléico, que é semelhante à molécula de DNA, porém com algumas diferenças na sua composição e estrutura. Tipos de RNA: RNA mensageiro (mRNA), RNA transportador (tRNA) e RNA ribossômico (rRNA).

Bibliografia

[1] AceDB. Sanger Institute. Disponível em: http://www.acedb.org. Acesso em: 27 set. 2005.

[2] Y. AKIYAMA, S. GOTO, I. UCHIYAMA, and M. KANEHISA. Cluster analysis and display of genome-wide expression patterns. *MIMBD'95: Second Meeting on the Interconnection of Molecular Biology Databases*, 1995.

[3] S. ALTSCHUL, W. GISH, W. MILLER, E. MYERS, and D. LIPMAN. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[4] S. ALTSCHUL, T.L. MADDEN, A.A. SCHAFFER, J. ZHANG, Z. ZHANG, W. MILLER, and LIPMAN D. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acid Research*, 25:3389–3402, 1997.

[5] R. V. ANDRADE. Análise do perfil transcricional e de genes diferencialmente expressos em micélio e levedura do fungo dimórfico *P. brasiliensis*. Tese de Doutorado em preparação, a ser defendida em março de 2006.

[6] APACHE. The Apache Software Foundation. Disponível em: http://httpd.apache.org. Acesso em: 11 out. 2005.

[7] Aspergillus fumigatus Genome Project. Disponível em: http://www.tigr.org/tdb/e2k1/afu1/. Acesso em: 7 dez. 2005.

[8] NCBI - National Center for Biotechnology Information. Genome Project - *Aspergillus fumigatus*. Disponível em: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=genomeprj&cmd=Retrieve&dopt=Overview&list_uids=9521. Acesso em: 18 out. 2005.

[9] Apache Axis. Disponível em: http://ws.apache.org/axis/news.html. Acesso em: 18 out. 2005.

[10] BIOFOCO. Núcleo de Informática do Centro-Oeste. Disponível em: http://www.biofoco.org. Acesso em: 16 jan. 2005.

[11] BIOPERL. Disponível em: http://www.bioperl.org. Acesso em: 11 out. 2005.

[12] B. BIRREN, G. FINK, and E. LANDER. A white paper for fungal comparative genomics. Disponível em: http://www.broad.mit.edu/annotation/fungi/fgi/FGI_02_whitepaper_2003.pdf. Acesso em: 28 out. 2005.

[13] BROAD Institute. Disponível em: http://www.broad.mit.edu/. Acesso em: 7 dez. 2005.

[14] BROAD Institute. *Aspergillus nidulans Database*. Disponível em: http://www.broad.mit.edu/annotation/fungi/aspergillus/. Acesso em: 27 out. 2005.

[15] S. K. CARD, J. D. MACKINLAY, and B. SHNEIDERMAN. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufman, San Francisco, CA, United States, 1999.

[16] D. CHAPPELL and T. JEWELL. *Java Web Services*. O'Reilly, United States, 2002.

[17] Consórcio Internacional de Seqüenciamento do Genoma Humano. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, Fev 2001.

[18] Consórcio Internacional de Seqüenciamento do Genoma Humano. Finishing the euchromatic sequence of the human genome. *Nature*, 431:931–945, Out 2004.

[19] DDBJ: DNA data Bank of Japan. Disponível em: http://www.ddbj.nig.ac.jp/. Acesso em: 6 dez. 2005.

[20] DOE. Joint Genome Institute. Disponível em: http://www.jgi.doe.gov. Acesso em: 16 jan. 2005.

[21] O. DUBUISSON. *ASN.1 - Communication between heterogeneous systems*. Morgan Kaufmann Publishers, Estados Unidos, 2000. Traduzido para o inglês por Philippe Fouquart.

[22] ECOCYC: Encyclopedia of *Escherichia coli K12* Genes and Metabolism. Disponível em: http://ecocyc.org/. Acesso em: 27 out. 2005.

[23] EMBL: European Molecular Biology Laboratory. Disponível em: http://www.embl.org/. Acesso em: 6 dez. 2005.

[24] L. FLOREA, M. MCCLELLAND, C. RIEMER, S. SCHWARTZ, and W. MILLER. Enterix 2003: visualization tools for genome alignments of enterobacteriaceae. *Nucleic Acids Research*, 31, No 13:3527–3532, 2003.

[25] FLYBASE - A Database of the *Drosophila* Genome. Disponível em: http://www.flybase.org/. Acesso em: 27 out. 2005.

[26] K. FRAZER, L. PACHTER, A. POLIAKOV, E. RUBIN, and I. DUBCHAK. Vista: computational tools for comparative genomics. *Nucleic Acids Research*, 1;32(Web Server issue):W273–W279, Jul 2004.

[27] GBROWSE. Generic Genome Browser. Disponível em: http://www.gmod.org/ggb/. Acesso em: 13 out. 2005.

[28] GMOD - Generic Model Organism Database. Disponível em: http://www.gmod.org/. Acesso em: 27 out. 2005.

[29] GRAMENE: A Comparative Mapping Resource for Grains. Disponível em: http://www.gramene.org/. Acesso em: 27 out. 2005.

[30] N. L. HARRIS. Genotator: A workbench for sequence annotation. *Genome Research*, 7:754–762, 1997.

[31] JDOM. Disponível em: http://www.jdom.org/. Acesso em: 18 out. 2005.

[32] M. KANEHISA and S. GOTO. Kegg: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*, 28:27–30, Jan 2000.

[33] KEGG: Kyoto Encyclopedia of Genes and Genomes. Disponível em: http://www.kegg.com/. Acesso em: 27 nov. 2005.

[34] S. LEWIS and et al. Apollo: a sequence annotation editor. *Genome Biology*, 3(12):research0082, Dez 2002.

[35] B. MCCORMICK, T.A. DEFANTI, and M.D. BROWN. *Visualization in Scientific Computing in Computer Graphics, volume 21.* ACM Press, November 1987.

[36] MGI - Mouse Genome Informatics. Disponível em: http://www.informatics.jax.org/. Acesso em: 27 out. 2005.

[37] B. MORGENSTERN, S. GOEL, A. SCZYRBA, and A. DRESS. AltAVisT: Comparing alternative multiple sequence alignments. *Bioinformatics*, 19, No 3:425–426, Fev 2003.

[38] MySQL. `Disponível em: https://www.mysql.com`. Acesso em: 18 out. 2005.

[39] NCBI - National Center for Biotechnology Information. `Disponível em: http://www.ncbi.nlm.nih.gov`. Acesso em: 17 jan. 2005.

[40] ONSA. Organization for Nucleotide Sequencing and Analysis. `Disponível em: http://watson.fapesp.br/onsa/Genoma3.htm`. Acesso em: 16 jan. 2005.

[41] W. R. PEARSON. Flexible sequence similarity searching with the FASTA3 program package. *Methods Mol. Biol.*, 132:185–219, 2000.

[42] W. R. PEARSON and D. J. LIPMAN. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA*, 85:2444–2448, 1988.

[43] PERL. `Disponível em: http://www.perl.org`. Acesso em: 11 out. 2005.

[44] Projeto Genoma Pb. `Disponível em: https://www.biomol.unb.br/cgi-bin/Pb/home/home.pl?projeto`. Acesso em: 18 out. 2005.

[45] Projeto Genoma Brasileiro. `Disponível em: http://www.brgene.lncc.br/`. Acesso em: 16 jan. 2005.

[46] E. T. RAY. *Learning XML*. O'Reilly, United States, Jan 2001.

[47] J. RUMBAUGH, I. JACOBSON, and G. BOOCH. *The Unified Modeling Language Reference Manual*. Addison Wesley, United States, 1999.

[48] SANGER. Sanger Institute. `Disponível em: http://www.sanger.ac.uk`. Acesso em: 27 set. 2005.

[49] J. C. SETUBAL and J. MEIDANIS. *Introduction to Computational Molecular Biology*. Brooks/Cole Publishing Company, Pacific Grove, CA, United States, 1997.

[50] SGD: *Saccharomyces* Genome Database. `Disponível em: http://www.yeastgenome.org/`. Acesso em: 27 out. 2005.

[51] A. J. G. SIMPSON and et al. The genome sequence of the plant pathogen Xylella fastidiosa. *Nature*, 406(6792):151–157, Julho 2000.

[52] SOAP. Disponível em: http://www.w3.org/TR/soap/. Acesso em: 18 out. 2005.

[53] I. SOMMERVILLE. *Engenharia de Software - Sexta Edição*. Addison Wesley, São Paulo, 2003.

[54] L. STEIN. Generic Genome Browser: A Tutorial. Disponível em: http://www.gmod.org/ggb/tutorial/tutorial.html. Acesso em: 11 out. 2005.

[55] TIGR. The Institute for Genomic Research. Disponível em: http://www.tigr.org. Acesso em: 16 jan. 2005.

[56] Universal Description, Discovery and Integration - UDDI. Disponível em: http://www.uddi.org. Acesso em: 18 out. 2005.

[57] J. C. VENTER and et al. The sequence of the human genome. *Science*, 291:1304–1351, Fev 2001.

[58] T. VOLK and T. MOSSMAN. Paracoccidioides brasiliensis, cause of paracoccidioidomycosis, aka South American Blastomycosis or Brazilian Blastomycosis. Disponível em: http://botit.botany.wisc.edu/toms_fungi/jan2005.html/. Acesso em: 6 dez. 2005.

[59] W3C. W3c - World Wide Web Consortium. Disponível em: http://www.w3.org/Protocols. Acesso em: 11 out. 2005.

[60] J. D. WATSON and F. H. C. CRICK. A structure for deoxyribose nucleic acid. *Nature*, 171:737–738, 1953.

[61] WORMBASE - The Biology and Genome of *C. Elegans*. Disponível em: http://www.wormbase.org/. Acesso em: 27 out. 2005.

[62] Web Service Description Language - WSDL. Disponível em: http://www.w3.org/TR/wsdl/. Acesso em: 18 out. 2005.

[63] Extensible Markup Language - XML. Disponível em: http://www.w3.org/XML/. Acesso em: 18 out. 2005.

[64] J. YANG, J. WANG, Z. YAO, Q. JIN, Y. SHEN, and R. CHEN. Genomecomp: a visualization tool for microbial genome comparison. *Journal of Microbiological Methods*, 54(3):423–426, Set 2003.