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

Sistema Multiagente para Anotação Manual em Projetos de Seqüenciamento de Genomas

Richardson Silva Lima

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Profª. Drª. Célia Ghedini Ralha

Brasília
2007

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

Coordenadora: Profª. Drª. Alba Cristina Magalhães de Melo

Banca examinadora composta por:

Profª. Drª. Célia Ghedini Ralha (Orientadora) – CIC/UnB
Profª. Drª. Ana Lucia Cetertich Bazzan – II/UFRGS
Profª. Drª. Maria Emilia Machado T. Walter – CIC/UnB

CIP – Catalogação Internacional na Publicação

Lima, Richardson Silva.

Sistema Multiagente para Anotação Manual em Projetos de Seqüenciamento de Genomas / Richardson Silva Lima. Brasília : UnB, 2007.
141 p. : il. ; 29,5 cm.

Tese (Mestre) – Universidade de Brasília, Brasília, 2007.

1. projeto de seqüenciamento de genomas, 2. anotação manual,
3. sistemas multiagente, 4. blackboard

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

Sistema Multiagente para Anotação Manual em Projetos de Seqüenciamento de Genomas

Richardson Silva Lima

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Profª. Drª. Célia Ghedini Ralha (Orientadora)
CIC/UnB

Profª. Drª. Ana Lucia Cetertich Bazzan
II/UFRGS

Profª. Drª. Maria Emilia Machado T. Walter
CIC/UnB

Profª. Drª. Alba Cristina Magalhães de Melo
Coordenadora do Mestrado em Informática

Brasília, 29 de Junho de 2007

Dedicatória

Este trabalho de mestrado dedico as pessoas mais importantes em toda minha formação: meus pais. José Ribamar Lima, meu pai, Rosa Maria Silva Lima, minha mãe, amo vocês!

Agradecimentos

Ao Senhor Deus em primeiro lugar, por todas as bençãos dadas por ele a mim.

À minha orientadora, Profª.Drª. Célia Ghedini Ralha, por toda as suas palavras de incentivo, pela confiança depositada em mim e sua paciência no ensino. Obrigado Profª. Célia, por todas as vezes em que sabiamente chamou minha atenção a relevância deste desafio. À Profª.Drª. Maria Emília M. T. Walter pela coorientação, principalmente por todo auxílio nos estudos relacionados ao mundo da Bioinformática, bem como da Biologia Molecular. À Profª.Drª. Natália Florêncio Martins (Embrapa/Cenargen) pela participação em minha banca de qualificação e pelos valiosos esclarecimentos em assuntos direcionados a Anotação de Genomas.

À Profª.Drª. Ana L. C. Bazzan pelo aceite ao convite para participar da banca avaliadora e pelas valiosas críticas e sugestões ao trabalho.

Ao Profº.Drº. Murilo Silva de Camargo por conceder-me fomento em forma de bolsa trabalho, a qual foi muito importante no suporte aos meus estudos. Ao Profº. Alan Montezano por todo seu apoio. À Rosa Amariles, secretária da pós-graduação, pela atenção e todo suporte dado durante o curso. Enfim, a todos os Professores do Mestrado pelos seus valiosos ensinamentos.

À minha namorada, Mayra Lucena, por toda compreensão aos muitos momentos de ausência, por todo companherismo nos momentos difíceis e por toda a paciência, carinho e amor nos momentos em que as dificuldades pareciam superiores. Ao meu irmão, Robson Diego, pela amizade. Aos meus familiares e a família da Mayra, obrigado por todo apoio.

Aos amigos Hugo Wruck Schneider e Anderson Gray F. Pereira por toda colaboração, companherismo e apoio no desenvolvimento deste trabalho. Ao amigo Rodrigo Carneiro M. Coimbra por esclarecimentos, opiniões sempre pertinentes e disponibilização de bibliotecas do *framework* Timina.

Aos amigos Antônio Emanuel, David Pereira, José Alves, Nathan Franklin, Rodrigo Veras e Rômulo Araújo, obrigado pela amizade e apoio desde nossas primeiras caminhadas no curso de Ciência da Computação na Universidade Federal do Piauí (UFPI). A todas as minhas amizades, muito obrigado!

À minha chefe no Serviço Federal de Processamento de Dados (SERPRO), Maria Lúcia Lopes, pela compreensão, apoio e por todas as liberações nos momentos que precisei. Aos amigos do pólo de desenvolvimento CEBSB, obrigado pelas palavras de apoio.

Resumo

Projetos de seqüenciamento de genomas obtêm as bases que compõem seqüências biológicas de um organismo, de tal forma que suas funções possam ser inferidas. As inferências de funções biológicas constituem uma das tarefas mais importantes destes projetos, as quais são realizadas na fase de anotação. Existem *softwares* e bases de dados que podem auxiliar os biólogos a realizar esta tarefa com uma maior acurácia e eficiência. Esta etapa é chamada de anotação automática. Os biólogos decidem as funções biológicas ou classificações das seqüências, com base nos seus conhecimentos biológicos. Esta etapa é chamada de anotação manual.

Sistemas Multiagente possuem uma arquitetura própria onde agentes inteligentes ou entidades autônomas interagem entre si de maneira cooperativa, compartilhando um objetivo comum ou agindo de acordo com seus objetivos individuais. Neste contexto, este trabalho propõe um sistema baseado na abordagem de Sistema Multiagente para apoiar o processo de anotação manual. A arquitetura do sistema provê a combinação de diferentes agentes, de tal forma que estes interajam entre si e com o ambiente, cooperando de forma a sugerir anotações manuais que deverão ser validadas pelos biólogos. Um protótipo, denominado *BioAgents*, foi desenvolvido sob uma arquitetura multiagente com abordagem *blackboard* a fim de validar a proposta. Como estudo de caso, aplicamos o *BioAgents* em dois projetos de seqüenciamento de genomas, cujo processamento foi realizado pelo Laboratório de Bioinformática do Instituto de Biologia da UnB, a saber: Projeto Genoma Funcional e Diferencial do *Paracoccidioides brasilienses* (Projeto Genoma Pb) e o Projeto Genoma Funcional e Genética Genômica de *Paullinia cupana* (Projeto Genoma Guaraná).

Palavras-chave: projeto de seqüenciamento de genomas, anotação manual, sistemas multiagente, blackboard

Abstract

Genome sequencing projects provide the basis to form the biological sequences of an organism, in such a away that their functions can be inferred. The biological function inferences constitute one of the most important tasks of these projects, which is carried through the annotation phase. There are softwares and databases that can help the biologists through with improved accuracy and efficiency. This task is called automatic annotation. The biologists decide the biological functions or the classification of the sequences based on their own biological knowledge. This task is called manual annotation.

Multiagent Systems provide an architecture where intelligent agents or autonomous entities interact with each other in a cooperative way, where they can share a common goal or act according to their own interests. This work presents a system based on the Multiagent System approach to support the process of manual annotation. The architecture of the system provides the combination of different agents, in such a way that these agents must interact among themselves and with the environment, suggesting annotations that will be later validated by the biologists. A prototype called BioAgents was developed under a multiagent architecture according to a blackboard approach in order to validate the proposal. Two study cases were developed applying the BioAgents in two genome sequencing projects of the Bioinformatics Laboratory of the Biology Institute of UnB - The *Paracoccidioides brasilienses* Functional and Differential Genome Project (Genome Pb Project) and The *Paullinia cupana* Functional and Genome Genetic Project (Genome Guaraná Project).

Keywords: genomes sequencing project, manual annotation, multiagent systems, blackboard

Sumário

Lista de Figuras **10**

Lista de Tabelas **12**

Capítulo 1 Introdução **15**
1.1 Contextualização 17
1.2 Motivação 18
1.3 Objetivos 19

Capítulo 2 Bioinformática **21**
2.1 Conceitos Básicos em Biologia Molecular 22
2.1.1 *DNA* e *RNA* 22
2.1.2 Genes, Cromossomos e Genomas 25
2.2 Seqüenciamento de Genomas 26
2.2.1 Seqüenciamento Estrutural 26
2.2.2 *EST* 28
2.2.3 Seqüenciamento Funcional 30
2.3 Comparação de Seqüências Genômicas 31
2.4 *Pipeline* de Bioinformática 32
2.4.1 Submissão, Montagem e Anotação 32
2.4.2 Exemplos de Anotação Automática 35
2.4.3 Sistemas de *Pipeline* 36
2.5 Ferramentas para Anotação 40
2.6 Bancos de Dados para Anotação 44

Capítulo 3 Sistemas Multiagente **52**
3.1 Agentes Inteligentes 53
3.2 Aspectos de SMA 55
3.2.1 Ambientes 57
3.2.2 Classificação 59
3.2.3 Arquitetura *Blackboard* 60
3.3 Desenvolvimento de SMA 61
3.3.1 Visão Geral sobre a Especificação *FIPA* 62
3.3.2 O *framework JADE* 65
3.4 Tópicos Relacionados ao SMA Proposto 69
3.4.1 Ontologia 69
3.4.2 Mineração de Dados 70

Capítulo 4 BioAgents **73**
4.1 Arquitetura proposta 73
4.2 O protótipo implementado 74
4.2.1 Arquitetura 75
4.2.2 Descrição das Camadas 77

Capítulo 5 Estudo de Caso e Discussão dos Resultados **87**
5.1 Anotação Manual nos Projetos Genoma Pb e Guaraná 87
5.2 Regras utilizadas 91
5.3 Resultados obtidos 93
5.3.1 Estudo de Caso com o Projeto Genoma Pb 93
5.3.2 Estudo de Caso com o Projeto Genoma Guaraná 95
5.3.3 Análise Geral e Outros Resultados 97
5.4 Trabalhos Correlatos 98

Capítulo 6 Conclusões e Trabalhos Futuros **103**

Apêndice A Código das Regras de Produção **106**

Apêndice B Publicações Obtidas **109**
B.1 Primeira publicação 109
B.2 Segunda publicação 111
B.3 Terceira publicação 121

Referências **125**

Glossário **138**

Lista de Figuras

2.1 (a) Estrutura molecular dos ácidos nucléicos *DNA* e *RNA*, e ligação entre bases nitrogenadas. (b) Estrutura molecular das pentoses: Ribose e Desoxirribose. (c) Dupla hélice de *DNA*, mostrando a ligação entre as bases nitrogenadas complementares, Adenina/Timina e Citosina/Guanina (adaptadas de [1, 2, 3]). 23
2.2 O Dogma Central da Biologia Molecular [136]. 24
2.3 (a) Quantidade de projetos de seqüenciamento completos e incompletos. (b) Distribuição de projetos entre os principais centros de seqüenciamento do mundo (adaptados de [4]). 27
2.4 Visão geral de como um banco *EST* é construído [53, 123]. 29
2.5 Exemplo de um alinhamento global. 32
2.6 Principais fases de um *pipeline* de seqüenciamento. 33
2.7 O formato *FASTA* [5]. 33
2.8 A anotação automática do Projeto Genoma Pb (adaptada de [123]). 36
2.9 A anotação automática do Projeto Genoma Guaraná. 37
2.10 A anotação automática do Projeto Genoma Anaplasma. 37
2.11 Figura referente ao *pipeline* para projetos genoma de *EST* no sistema *BioNotes* [101]. 39
2.12 (a) Dimensões, em número de seqüências e pares de bases, das bases de dados, de acordo com os útimos *releases* disponibilizados: *GenBank* (*release* 158 - fevereiro de 2007), *EMBL* (*release* 90 - março de 2007) e *DDBJ* (*release* 69 - março de 2007). (b) Interação entre as bases de dados do consórcio *INSDC*. 45
2.13 Crescimento exponencial apresentado pelo *GenBank*. 46
2.14 Crescimento exponencial apresentado pelo *EMBL*. 47
2.15 Crescimento exponencial apresentado pelo *DDBJ*. 47
2.16 Porcentagens das bases *Pfam-A* e *Pfam-B* em relação as 8957 famílias em *Pfam* (adaptada de [6]). 49

3.1 Arquitetura clássica de um agente [129]. 53
3.2 Ambiente divido em três camadas (adaptado de [154]). 58
3.3 Esquema simples de uma arquitetura *blackboard* (adaptada de [70]). 62
3.4 Modelo de referência *FIPA* para a plataforma de agentes (adaptada de [66]). 64
3.5 Arquitetura do *framework JADE* (adptada de [55]). 67
3.6 Plataforma *JADE* distribuída em vários *containers* (adaptada de [56]). 68

4.1 Arquitetura proposta composta por três camadas. 74
4.2 Arquitetura do *BioAgents* composta por 3 camadas. 76
4.3 *Screenshot* da tela de execução e do *sniffer* (módulo disponibilizado pelo *framework JADE*) mostrando as mensagens trocadas entre agentes do protótipo *BioAgents*. 78
4.4 Fluxo de comunicação entre os agentes GR e agentes ANL. 79
4.5 Conteúdo e estrutura das sugestões enviadas pelos agentes ANL aos agentes GR. 81
4.6 Exemplo de um alinhamento obtido com a execução do programa *BLASTX*. 81
4.7 Ciclo de vida e o fluxo de troca de mensagens entre os agentes do protótipo *BioAgents*, considerando o exemplo de sugestões a partir da análise de resultados do *software BLAST*. 82
4.8 Estrutura do vocabulário *BioOntology*. 84
4.9 Estrutura do vocabulário *CommunicationOntology*. 84
4.10 Relacionamento entre as classes que compõem as sugestões fornecidas pelo *BioAgents*. 85
4.11 Diagrama de classes apresentando a relação entre as classes *RequestConfig*, *RequestAlgorithm* e *RequestFile*. 86

5.1 Arquivo de saída (em formato *html*), armazenado em um banco *PostgreSQL*, contendo os resultados originais obtidos com a execução do programa *BLASTX* sobre a base de proteínas *nr*. 89
5.2 Página *Web* utilizada para a anotação manual do Projeto Genoma Pb. 89
5.3 Página *Web* utilizada para a anotação manual do Projeto Genoma Guaraná. 90
5.4 Diagrama de atividades referente ao fluxo de instruções das regras de produção. 92
5.5 Distribuição do total de grupos do Projeto Genoma Pb. 94
5.6 Distribuição do total de grupos do Projeto Genoma Pb não anotados e os anotados manualmente. 94
5.7 Distribuição das sugestões fornecidas pelo *BioAgents* ao Projeto Genoma Pb. 95
5.8 Distribuição do total de grupos do Projeto Genoma Guaraná. . . 96
5.9 Distribuição do total de grupos do Projeto Genoma Guaraná não anotados e os anotados manualmente. 96
5.10 Distribuição das sugestões fornecidas pelo *BioAgents* ao Projeto Genoma Guaraná. 97

B.1 A arquitetura em três camadas do sistema *BioAgents*. 115
B.2 *Screenshot* da tela de execução e do *sniffer* dos agentes do *BioAgents* no framework *JADE*. 117
B.3 Conjunto de regras *Jess* para análise de saídas *BLAST* e *FASTA*. 118
B.4 BioAgents Architecture. 122

Lista de Tabelas

2.1 Conjunto de programas *BLAST*. 42

3.1 Exemplos de SMA e seus ambientes (adaptada de [129]). 57

5.1 Resultados dos testes com o Projeto Genoma Pb (adaptada de [107]). 95
5.2 Resultados dos testes com o Projeto Genoma Guaraná. 97
5.3 Comparação entre os trabalhos correlatos discutidos. 102

B.1 Resultados do *BioAgents* utilizando dados do Projeto Genoma Pb. 119

Lista de Abreviaturas

AID : *Agent Identifiers* (Identificadores de Agentes).

BLAST : *Basic Local Alignment Search Tool* (Ferramenta para Busca de Alinhamentos Locais).

COG : *Clusters of Orthologous Groups of proteins* (Agrupamento de Grupos Ortólogos de Proteínas).

DDBJ : *DNA DataBank of Japan* (Banco de Dados de DNA do Japão).

DNA : *Deoxyribonucleic Acid* (Ácido Desoxirribonucléico).

EBI : *European Bioinformatics Institute* (Instituto Europeu de Bioinformática).

EMBL : *European Molecular Biology Laboratory* (Laboratório Europeu de Biologia Molecular).

EST : *Expressed Sequences Tags* (Rótulos de Seqüências Expressa).

FIPA : *Foundation for Intelligent Physical Agents* (Fundação para Agentes Físicos Inteligentes).

GO : *Gene Ontology* (Ontologias de Genes).

GOLD : *Genomes Online Databases* (Bases de Dados *Online* de Genomas).

HMMs : *Hidden Markov Models* (Modelos Ocultos de Markov).

IA : Inteligência Artificial.

IAD : Inteligência Artificial Distribuída.

IEEE : *Institute of Electrical and Electronics Engineers* (Instituto de Engenharia Elétrica e Eletrônica).

INSDC : *International Nucleotide Sequence Database Collaboration* (Colaboração Internacional de Bases de Seqüências Nucleotídicas).

JADE : *Java Agent DEvelopment Framework* (*Framework Java* para Desenvolvimento de Agentes).

JESS : *Java Expert System Shell* (Programa interpretador de comandos para Sistemas especialistas em *Java*).

JGI : *Joint Genome Institute* (Instituto Comum do Genoma).

JVM : *Java Virtual Machine* (Máquina Virtual Java).

KEGG : *Kyoto Encyclopedia of Genes and Genomes* (Enciclopédia *Kyoto* de Genes e Genomas).

MCs : Modelos de Covariância.

NCBI : *National Center for Biotechnology Information* (Centro Nacional para Informações de Biotecnologia, USA).

NIH : *National Institutes of Health* (Instituto Nacional de Saúde).

ORFs : *Open Reading Frames* ("Regiões Potencialmente Codificantes").

PSORT : *Prediction of Protein Sorting Signals and Localization Sites In Amino Acid Sequences* (Predição da Localização e Classificação de Sinais em Sítios de Proteínas em Seqüências de Aminoácidos.).

RMI : *Remote Method Invocation* (Invocação Remota de Métodos).

RNA : *Ribonucleic Acid* (Ácido Ribonucléico).

SMA : Sistema MultiAgente.

SMART : *Simple Modular Architecture Research Tool* (Ferramenta Modular Simples para Pesquisa de Arquitetura).

TIGR : *The Institute for Genomic Research* (Instituto para Pesquisas Genômicas).

UML : *Unified Modeling Language* (Linguagem de Modelagem Unificada).

USDE : *United States Departament of Energy* (Departamento Norte Americano de Energia).

WTSI : *Wellcome Trust Sanger Institute* (Instituto Britânico Sanger *Wellcome Trust*).

Capítulo 1

Introdução

Em 1953, os pesquisadores James Watson e Francis Crick, baseados em vários trabalhos anteriores sobre o *Deoxyribonucleic Acid (DNA)*, publicaram um trabalho na revista *Nature* intitulado *Molecular Structure of Nucleic Acids- A Structure for Deoxyribose Nucleic Acid* [151]. Com esta publicação, era anunciado ao mundo a descoberta da estrutura do *DNA*. Desde essa época, a comunidade científica vem dispendendo grandes esforços com o objetivo de compreender melhor a estrutura e o funcionamento da biologia molecular dos seres vivos, sendo que estes esforços têm sido os propulsores dos avanços em técnicas da Biologia Molecular e, mais recentemente, da Biologia Computacional, também conhecida como Bioinformática.

No início da década de 90, foi iniciado o Projeto Genoma Humano, que visava mapear e seqüenciar, por completo, o genoma humano. Este projeto foi concluído em 2001 [150, 99], e apresentou o genoma humano com 3 bilhões de pares de bases e aproximadamente 30.000 genes, tornando-se um marco importante para o desenvolvimento das pesquisas genéticas em todo o mundo.

O Projeto Genoma Humano e inúmeros outros projetos de seqüenciamento de genomas surgidos em todo o mundo propiciaram grandes e rápidos avanços em técnicas da Biologia Molecular e Bioinformática [108]. Assim, desde a década de 90, podemos observar um crescimento exponencial no volume de dados gerados pelos diversos projetos de seqüenciamento de genomas. Em relação ao gerenciamento e análise destes dados, a área de Computação tem desenvolvido técnicas e *softwares* que apoiam o esforço dos biólogos no armazenamento e análise dos dados gerados nestes projetos.

No desenvolvimento de um projeto de seqüenciamento de genomas, existem diversas tarefas a serem realizadas, no entanto, destacamos a compreensão e interpretação dos dados resultantes do processo de seqüenciamento, que usa basicamente algoritmos para comparação entre seqüências biológicas. A execução desta tarefa é conhecida como processo de anotação.

Comparar seqüências é uma das operações mais básicas em Bioinformática e consiste em buscar regiões semelhantes entre seqüências. Seqüências bastante similares entre si geralmente implicam similaridades estruturais e funcionais entre organismos.

A comparação de seqüências é utilizada em projetos de seqüenciamento de genomas comparando as seqüências geradas no projeto com seqüências de bancos

de dados privados e/ou públicos (como o *GenBank* [57]) que tiveram suas funções biológicas previamente determinadas.

Algoritmos como os de Needleman-Wunsch [119] e Smith-Waterman [139], ambos baseados na técnica de programação dinâmica, são clássicos para a resolução do problema de comparação de seqüências. Outros métodos de comparação de seqüências, baseados em heurísticas, são encontrados nas ferramentas *Basic Local Alignment Search Tool* (BLAST) [43] e o *FASTA* [109, 124]. A comparação de seqüências utilizando estes métodos computacionais é conhecida como anotação automática (*in silico*[1]).

Com base nos resultados da anotação automática, os biólogos têm que decidir as funções biológicas a serem associadas às seqüências geradas no projeto. Esta tarefa é conhecida como anotação manual.

A anotação manual garante acurácia e completude ao entendimento dos dados biológicos e tem como objetivo decidir funções de genes das seqüências geradas em um projeto de seqüenciamento, usando funções conhecidas de genes similares disponibilizados em bases de dados biológicas [102, 75]. É importante observar que há projetos de seqüenciamento no qual o processo de anotação estende-se a uma etapa conhecida como re-anotação. De acordo com [118], a re-anotação consiste em obter informações adquiridas da anotação original, revisando-as e comparando-as com novos modelos e dados avaliados, a fim de se obter novas características e informações sobre as seqüências originalmente anotadas. Enfim, re-anotações podem ser feitas ao longo do tempo não apenas pela disponibilidade de técnicas computacionais mais recentes e confiáveis, mas também devido a erros de análise (por exemplo, anotações erradas de funções relacionadas a proteínas e genes diferentes anotados com o mesmo nome) que podem ocorrer na tarefa de anotação manual [90, 148].

Este trabalho está relacionado à anotação manual, tarefa inserida entre a anotação automática e a re-anotação (quando esta é executada). A hipótese do trabalho é o fato de que os biólogos decidem - devido ao conhecimento tácito, adquirido com a experiência profissional que estes possuem acerca da anotação de genomas - a anotação de uma determinada seqüência. No entanto, na tarefa de anotação manual, há operações - como a anotação de produtos biológicos (por exemplo, proteínas) - que podem ser automatizadas a fim de reduzir o trabalho dos biólogos, auxiliando-os na conclusão desta etapa. O foco deste trabalho é utilizar a abordagem de Sistemas MultiAgente (SMA) [143, 152, 157, 129, 112] na tarefa de anotação manual, propondo um mecanismo automático que provê anotações manuais aos projetos de seqüenciamento de genomas.

Observando este contexto, é proposta uma arquitetura baseada em SMA com objetivo de auxiliar os biólogos durante a tarefa de anotação manual, permitindo que a execução desta tarefa seja realizada com uma maior acurácia. A arquitetura proposta combina diferentes agentes com algoritmos em mineração de dados [81, 156] e ontologias específicas [92, 50], interagindo entre si e com o ambiente, cooperando de forma a sugerir anotações manuais que deverão ser validadas pelos biólogos [105]. Para validar esta proposta, este trabalho apresenta a implementação do protótipo denominado *BioAgents* [107, 106]. Este protótipo

[1] Anotação realizada por processamento computacional.

está sustentado em uma arquitetura multiagente conforme uma abordagem *blackboard* [121, 70, 71], na qual diferentes agentes trocam mensagens por meio de vocabulários comuns de comunicação e representação de informações biológicas encontradas nas bases de dados utilizadas pelo *BioAgents*. O ambiente de comunicação, bem como seus protocolos, estão sob gerenciamento do *Java Agent Development Framework (JADE)* [55, 56]. O mecanismo de raciocínio utilizado pelo *BioAgents* baseia-se em regras de produção, as quais foram implementadas utilizando a ferramenta *Java Expert System Shell (JESS)* [93]. O *BioAgents* sugere uma anotação manual com base no conjunto de resultados obtidos da anotação automática de uma particular seqüência do genoma analisado. Estas sugestões deverão ser validadas pelos biólogos.

1.1 Contextualização

Existem diversos centros especializados no seqüenciamento de genomas espalhados pelo mundo, tanto na esfera pública como privada. Como exemplo, podemos citar o *The Institute for Genomic Research* (*TIGR*) [7] , o *Sanger Centre-Wellcome Trust Sanger Institute* (*SC-WTSI*) [8], o *European Molecular Biology Laboratory-European Bioinformatics Institute* (*EMBL-EBI*) [67], o *National Center for Biotechnology Information* (*NCBI*) [9], dentre outros. No Brasil, em 1997, a Fundação de Amparo a Pesquisa do Estado de São Paulo (FAPESP) foi a primeira instituição governamental a apoiar o desenvolvimento de projetos de seqüenciamento de genomas. Foi criado um instituto virtual conhecido como rede *Organization for Nucleotide Sequencing and Analysis* (*ONSA*).

O consórcio *ONSA*, cujo modelo de formação privilegiou a criação de redes de cooperação entre centros de pesquisa, integrou neste primeiro projeto diversos laboratórios paulistas para seqüenciamento de *DNA* em larga escala, sendo que o suporte na área de Bioinformática encontrava-se centralizado no Instituto de Computação da Universidade Estadual de Campinas (Unicamp). O primeiro resultado importante desse instituto, bem como o seu reconhecimento internacional, ocorreu com a publicação do genoma da bactéria *Xylella fastidiosa*, agente etiológico, ou seja, o causador da *Citrus Variegated Chlorosis* (*CVC*), mais conhecida como praga do amarelinho. Esta doença é capaz de destruir lavouras de laranja, provocando perdas econômicas consideráveis na produção e, conseqüentemente, na exportação do suco de laranja. Em âmbito mundial, até o ano de 2001, o *Xylella-CVC* foi o primeiro fitopatógeno seqüenciado e anotado, o que resultou em um marco histórico para a comunidade científica brasileira [137].

Após o projeto de seqüenciamento da bactéria *Xylella fastidiosa*, a FAPESP investiu em outros projetos, tais como o do mapeamento do genoma da cana-de-açúcar; do câncer humano, realizado em parceria com o *Ludwig Institute for Cancer Research*; seqüenciamento do genoma funcional do parasita *Schistosoma mansoni*, causador da esquistossomose, popularmente conhecida como barriga d'água; entre outros projetos.

O governo federal, por meio do Ministério da Ciência e Tecnologia (MCT) e do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) no final do ano de 2000, objetivando ampliar a competência nas atividades de

pesquisa e manipulação de genoma, lançou o Projeto Genoma Brasileiro. Este projeto teve a participação de 25 laboratórios de Biologia Molecular, distribuídos pelas 5 regiões do país. Como resultado, o projeto já seqüenciou o genoma da bactéria *Chromobacterium violaceum* [149]. O MCT e o CNPq induziram projetos em redes regionais de pesquisa, as quais são: Rede Genoma do Estado de Minas Gerais; Rede Genoma Nordeste (ProGeNe); Programa de Implantação do Instituto de Biologia Molecular do Paraná; Programa Genoma do Estado do Paraná (GenoPar); Programa de Implantação da Rede Genoma do Estado do Rio de Janeiro (RioGene); Ampliação da Rede de Genômica no Estado da Bahia; Rede da Amazônia Legal de Pesquisas Genômicas (REALGENE); Programa de Investigação de Genomas Sul (PIGS); e Rede Genoma Centro-Oeste.

A Rede Genoma Centro-Oeste foi criada integrando instituições de ensino e pesquisa em Biologia Molecular dos seguintes estados: Distrito Federal, Goiás, Mato Grosso e Mato Grosso do Sul. O organismo seqüenciado foi o fungo *Paracoccidiodes brasiliensis* (Projeto Genoma Funcional e Diferencial do *Paracoccidioides brasilienses* - Projeto Genoma Pb) [82, 83, 61]. Foi criado o Laboratório de Bioinformática no Instituto de Biologia da UnB. Atualmente, este laboratório é responsável pela bioinformática do Projeto Genoma Funcional e Genética Genômica de *Paullinia cupana* (Projeto Genoma Guaraná) e do Projeto Análise Genômica Estrutural da bactéria *Anaplasma marginale* (Projeto Genoma Anaplasma).Os projetos genoma Pb, Guaraná e Anaplasma [10] serão descritos neste trabalho .

Em 2002, foi criada na região Centro-Oeste a Rede de Pesquisa e Desenvolvimento em Bioinformática do Centro-Oeste (BIOFOCO) que uniu a UnB, a UCB e Embrapa-Recursos Genéticos e Biotecnologia. O projeto BIOFOCO [11] teve seu desenvolvimento dividido em duas fases. A primeira fase, teve como objetivo geral integrar as instituições de pesquisa e ensino desta região e oferecer apoio aos grupos de pesquisa genômica e proteômica, através da troca de conhecimentos, ferramentas, sistemas e capacitação de técnicos e pesquisadores [86]. Já a segunda fase, objetivou desenvolver um ambiente de *grid computing* (computação em grade), que consiste em compartilhar os recursos computacionais de diversas máquinas distribuídas entre instituições de forma a executar aplicações em larga escala. Nesta segunda fase, alguns trabalhos foram desenvolvidos e outros encontram-se em desenvolvimento [140, 127].

1.2 Motivação

A enorme quantidade de dados gerada pelos projetos de seqüenciamento resultou em demandas que estimulam os pesquisadores a desenvolverem técnicas e *softwares* cada vez mais eficazes para o auxílio e otimização da análise, interpretação e entendimento do significado biológico desses dados.

Neste contexto, a Bioinformática possui como um dos seus grandes desafios auxiliar os biólogos a minerar os dados, converter os dados experimentais em informações biologicamente relevantes, perceber, explorar e testar suas idéias rapidamente, acessar rapidamente as anotações armazenadas nas bases de dados externas, executar *softwares* de apoio a anotação e obter rapidamente as informações geradas por eles, analisar as anotações existentes e gerar novas anotações manu-

almente [102].

Portanto, a Bioinformática apresenta-se como um domínio que necessita lidar com inúmeras e volumosas bases de dados heterogêneas, bem como inúmeros *softwares*. Desta forma, verifica-se a necessidade de prover aos biólogos ferramentas que lhes proporcionem ganho no tempo dispendido para realizar tarefas como a de anotação manual.

Considerando as várias tecnologias e metódos utilizados em Bioinformática, a área de Inteligência Artificial (IA), com ênfase na abordagem de SMA, pode ser usada como técnica computacional para o desenvolvimento e automatização das tarefas de anotação automática, anotação manual e re-anotação em projetos de seqüenciamento de genomas. Sua aplicação na área de Bioinformática é interessante por essa área ser um domínio que apresenta características como bases de dados heterogêneas e descentralizadas, ambiente dinâmico (por exemplo, novos tipos de dados, novas bases de dados e alteração das bases existentes são situações que podem ocorrer constantemente), a anotação poder ser realizada de forma independente, entre outras.

Vários trabalhos na área de Bioinformática utilizam técnicas de IA, através do uso de abordagens distintas como a de SMA, Ontologias, Mineração de Dados (*Data Mining*) e Aprendizagem de Máquina (*Machine Learning*). Essas abordagens têm sido aplicadas desde a busca e comparação de genomas até a atualização e/ou correções de seqüências originalmente anotadas, objetivos das anotações automática e re-anotação. Assim, a grande maioria destes trabalhos possuem em comum o apoio ao processo de anotação. Porém, pelo tanto que sabemos, os trabalhos da literatura que aplicam a metodologia de SMA para anotação de genomas não são aplicados exclusivamente na tarefa de anotação manual.

1.3 Objetivos

O objetivo principal deste trabalho é aplicar a abordagem de SMA, através de uma arquitetura própria, para auxiliar os biólogos a realizarem a tarefa de anotação manual em projetos de seqüenciamento de genomas, tendo como produto uma arquitetura implementada e validada por estudos de caso. Os objetivos específicos são:

- Desenvolver um protótipo, baseado em uma arquitetura multiagente com utilização de regras de produção, com o objetivo de prover sugestões de anotações manuais relacionadas as *Expressed Sequences Tags* (*EST*) identificadas nos projetos genoma Pb e Guaraná. Estas sugestões devem ser confirmadas pelos biólogos;

- Comparar as sugestões do protótipo desenvolvido com as anotações manuais já realizadas para os projetos genoma Pb e Guaraná, a fim de validá-lo;

Essa dissertação está organizada em seis capítulos. No Capítulo 2 são descritos tópicos como conceitos de Biologia Molecular e temas relevantes de Bioinformática. No Capítulo 3 são abordados tópicos relacionados a conceitos e ao desenvolvimento de SMA, bem como uma breve explanação de temas relacionados ao SMA Proposto. No Capítulo 4 apresentaremos a arquitetura proposta

nesta dissertação, bem como as características e a arquitetura do protótipo desenvolvido. No Capítulo 5 são descritos tópicos relacionados ao estudo de caso, discussão dos resultados e trabalhos correlatos. Por fim, no Capítulo 6 são descritas as conclusões, bem como possíveis trabalhos futuros.

Capítulo 2

Bioinformática

Desde o início do Projeto Genoma Humano em 1990, a Bioinformática surge como uma área de pesquisa com característica interdisciplinar, que visa estudar, desenvolver e aplicar técnicas e ferramentas computacionais ao processamento das informações associadas às seqüências biológicas e estruturas celulares estudadas pela Biologia Molecular [123, 68, 94]. Estas informações são obtidas através dos diversos e volumosos bancos de dados biológicos existentes no mundo. Grande parte destes bancos possuem dados incompletos e/ou com a presença de ruídos, o que traz à Computação o desafio de aprimorar a correção e eficiência dos algoritmos desenvolvidos. Existem dois meios para se alcançar os grandes objetivos da Bioinformática:

- Prover um meio de gerenciar e organizar os dados biológicos de forma a facilitar o acesso as informações e a submissão de novos dados a medida que estes são gerados;

- Desenvolver ferramentas de manipulação, processamento e análise dos dados armazenados, para prover aos especialistas uma otimização do tempo de trabalho e um ganho na qualidade da interpretação dos resultados.

São tarefas típicas executadas dentro da Bioinformática: comparação entre seqüências; encontrar todos (ou específicos) genes e proteínas de um dado genoma; inferir tipos e funções de proteínas em uma dada seqüência de aminoácidos; determinar regiões em estruturas e proteínas onde moléculas (por exemplo, de medicamentos) podem ser ativadas [68].

Para a realização destas tarefas, usualmente são investigadas as seqüências homólogas que já tiveram suas estruturas e funções comprovadas. Homologia entre duas seqüências (ou estruturas) sugere que elas possuem uma ligação em comum (uma história evolucionária em comum; compartilham um mesmo ancestral) e possivelmente a mesma função. Homologia pode ser computacionalmente expressa pela similaridade [53]. Duas seqüências são consideradas similares, quando trechos destas seqüências são "aproximadamente iguais", isto é, as duas seqüências têm exatamente os mesmos caracteres (por exemplo, bases nitrogenadas), com poucas exceção de caracteres diferentes, ou inserções e remoções de caracteres de uma das seqüências em relação à outra. Se existir similaridade

alta entre duas seqüências, observa-se um indício que tais seqüências podem ser homólogas [68].

Nesse capítulo são apresentados conceitos básicos de Biologia Molecular na Seção 2.1. Uma breve explanação de como comparar duas seqüências é feita na Seção 2.3. Duas técnicas usadas em projetos de seqüenciamento de genomas, bem como uma breve explanação do conceito de *EST*, são feitos na Seção 2.2. O *pipeline* de projetos de seqüenciamento de genomas (fases de submissão, montagem e anotação) são apresentados na Seção 2.4. *Softwares* utilizados na anotação automática são descritos na Seção 2.5. Por fim, a descrição de bases de dados utilizadas processo de anotação é feita na Seção 2.6.

2.1 Conceitos Básicos em Biologia Molecular

Nessa seção apresentamos conceitos de Biologia Molecular que serão utilizados nesta dissertação.

2.1.1 *DNA* e *RNA*

Como repositório da informação genética, o *DNA* ocupa uma posição única e central entre as macromoléculas biológicas. As seqüências nucleotídicas do *DNA* descrevem as estruturas primárias de todas as seqüências de *Ribonucleic Acid (RNA)*, proteínas celulares e das enzimas que são capazes de controlar o tipo e a quantidade de todos os componentes celulares. Assim, o *DNA* determina características genéticas de um ser vivo [100, 69].

O *DNA* é um dos dois tipos de ácidos nucléicos encontrados dentro da célula de um organismo. O *DNA* é composto por nucleotídeos, onde cada nucleotídeo é composto por três partes: uma pentose (açúcar) denominada desoxirribose, um grupo fosfato e uma base nitrogenada, que pode ser Adenina(A), Citosina(C), Guanina(G) ou Timina(T) [52]. A Figura 2.1 exibe a estrutura de uma molécula de *DNA*, que consiste de duas cadeias que se enrolam ao redor do mesmo eixo formando uma dupla hélice, da direção $5'$ para $3'$, isto é, direção do carbono-5 para o e carbono-3 da pentose.

As bases A, G, C e T de ambas as fitas encontram-se ligadas dentro da dupla hélice. As fitas são complementares entre si, ou seja, toda vez que aparecer uma Adenina numa cadeia, Timina será encontrada na outra, onde se encontrar Guanina numa cadeia, encontrar-se-á Citosina na outra. As duas cadeias ou fitas da hélice são antiparalelas, isto é, suas ligações 5' e 3' correm em direções opostas [62, 69].

O outro tipo de ácido nucléico é o *RNA* (Figura 2.1), que apresenta estrutura celular similar a do *DNA*. A diferença está na pentose que compõe o nucleotídeo, que é uma ribose ao invés de uma desoxirribose, e na existência do nucleotídeo que possui uma base nitrogenada chamada Uracila (U), substituindo o nucleotídeo que possui a Timina [62, 52].

Várias classes de *RNAs* são encontradas na célula, cada uma com uma função distinta. Os *RNA* ribossômicos ou *ribossomic RNA* (*rRNA*) são componentes estruturais dos ribossomos, que são grandes complexos envolvidos no processo

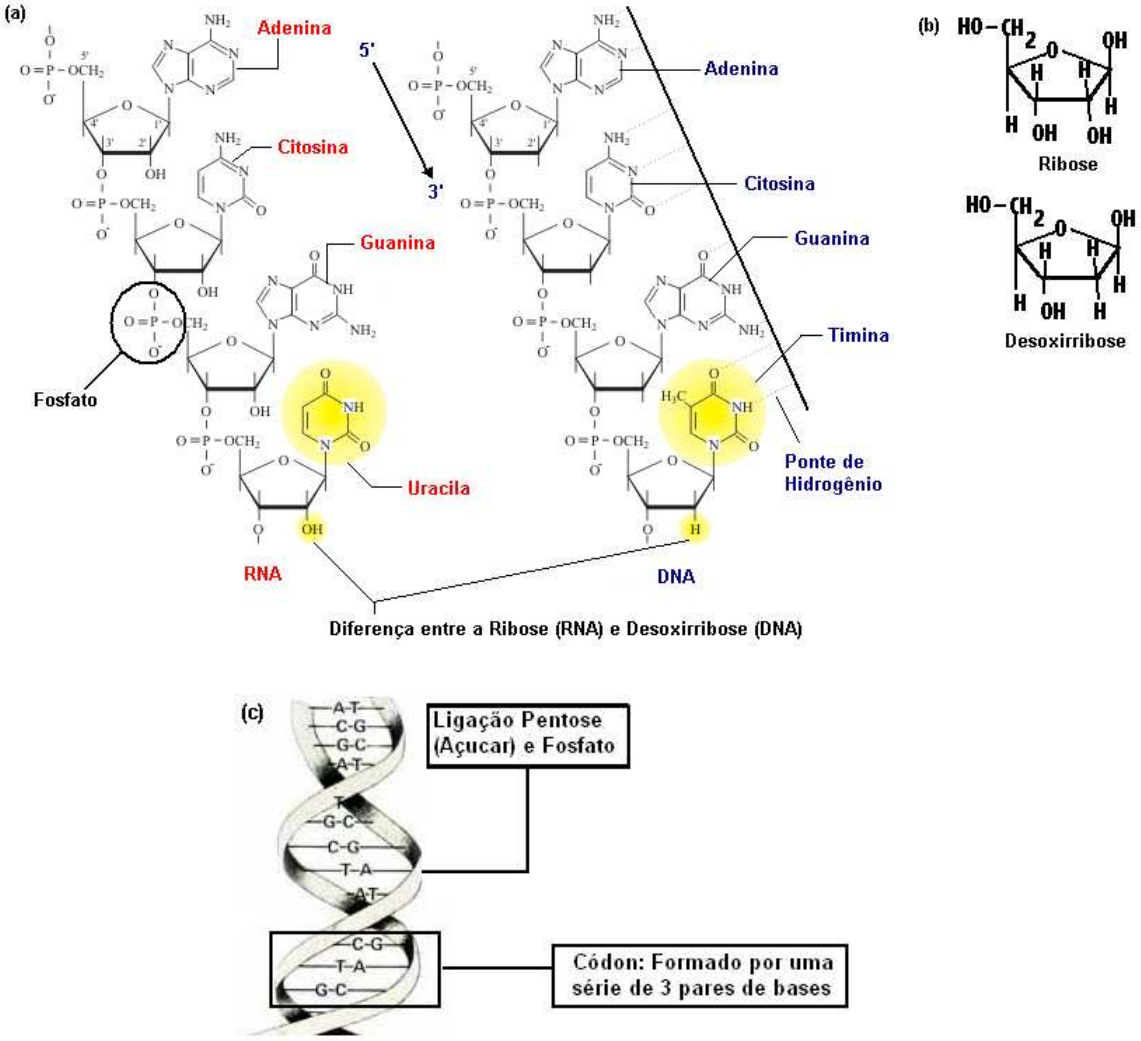


Figura 2.1: (a) Estrutura molecular dos ácidos nucléicos *DNA* e *RNA*, e ligação entre bases nitrogenadas. (b) Estrutura molecular das pentoses: Ribose e Desoxirribose. (c) Dupla hélice de *DNA*, mostrando a ligação entre as bases nitrogenadas complementares, Adenina/Timina e Citosina/Guanina (adaptadas de [1, 2, 3]).

da síntese de proteínas. Os *RNA* mensageiros ou *messenger RNA* (*mRNA*) são ácidos nucléicos que transportam a informação de um ou de uns poucos genes até o ribossomo, onde as proteínas correspondentes serão sintetizadas. Os *RNA* de transferência ou *transfer RNA* (*tRNA*) são moléculas adaptadoras que traduzem a informação presente no *mRNA* numa seqüência específica de aminoácidos [62].

O Dogma Central da Biologia Molecular

O dogma central define o paradigma da Biologia Molecular, em que a informação é perpetuada através da replicação do *DNA* [136]. Esta replicação é o processo de cópia do *DNA* pai para formar as moléculas filhas, tendo seqüências nucleotídicas idênticas, e é traduzida através de dois processos: transcrição - que converte a informação do *DNA* em uma forma mais acessível (uma fita de *RNA* complementar); e tradução - que converte a informação contida no *RNA* em proteínas. A Figura 2.2 ilustra o dogma central, de acordo com as fases de replicação, transcrição e tradução.

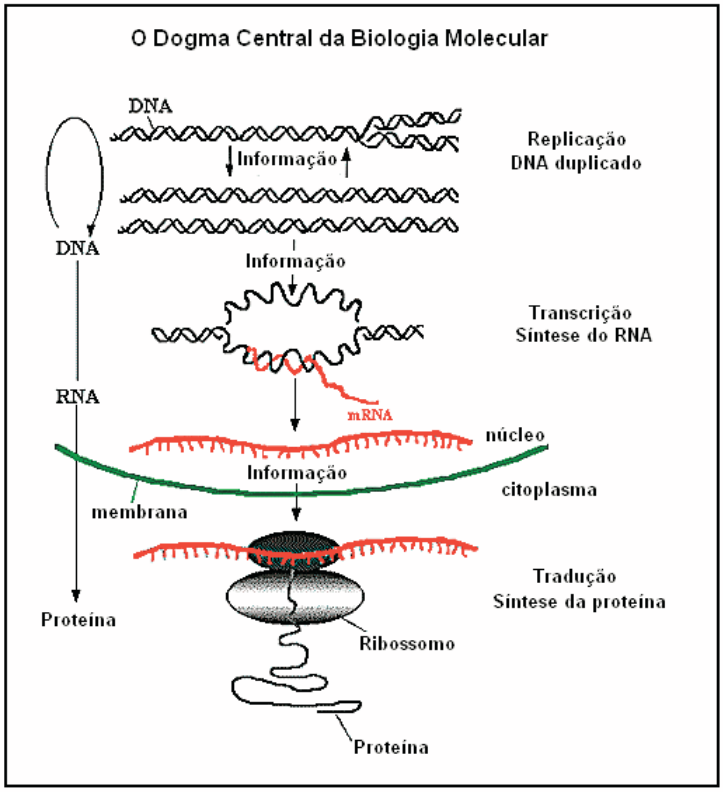


Figura 2.2: O Dogma Central da Biologia Molecular [136].

As proteínas são seqüências de aminoácidos produzidas no processo de tradução da informação codificada no *mRNA*. Para especificar uma proteína, basta especificar sua seqüência de aminoácidos. Os aminoácidos são definidos por unidades informacionais no *mRNA* chamadas de códons. Os códons para os aminoácidos são constituidos de trincas nucleotídicas específicas. A tradução requer moléculas

adaptadoras, os *tRNA*, que reconhecem códons e inserem aminoácidos em suas posições seqüenciais apropriadas no polipeptídeo [69].

Na síntese de proteínas, uma *reading frame* (fase de leitura) é uma das três possíveis formas de agrupar as bases para formar códons em uma seqüência de *DNA* ou *RNA*. Considerando o fragmento de seqüência GGATCAGCGC, por exemplo, uma possível fase de leitura seria GGA, TCA, GCG, ignorando a última base C, onde foram formados códons a partir da primeira base; outra fase de leitura seria feita ignorando-se a primeira base G e agrupando as demais bases nos seguintes códons GAT, CAG, CGC. Uma terceira fase possível ignoraria as duas bases GG formando ATC, AGC, e desprezando as duas bases finais GC. Dessa forma, existem três possíveis fases de leitura da seqüência de bases do *DNA*, iniciando na primeira, segunda ou terceira letras da seqüência. A partir da quarta letra, as fases de leituras são iguais a uma das três primeiras fases, com um ou mais códons a menos [134, 69].

Uma fase aberta de leitura ou *Open Reading Frame* (*ORF*), em seqüências de *DNA*, é uma seqüência que começa no códon inicial (geralmente ATG) e possui comprimento múltiplo de três, sendo completamente mapeada em códons sem precisar ignorar nenhuma base no final da seqüência [134, 62]. Procurar por *ORF* que iniciam com um códon ATG é uma forma de localizar genes [62].

2.1.2 Genes, Cromossomos e Genomas

Os organismos vivos apresentam características observáveis tais como cor dos olhos, cor da pele, comportamento, entre outras. À essas características, dá-se o nome de fenótipo, que é determinado pela interação entre genótipo do organismo e o meio em que ele vive. O genótipo é o conjunto de informações contidas no material genético de um organismo. Estas informações determinam como o organismo será construído e mantido. Elas são replicadas a cada divisão celular e podem ser herdadas no momento da reprodução [44, 52].

Os **genes** são as principais unidades de informação biológica contidas no material genético de um organismo. Um gene armazena as instruções para síntese de proteínas. Na maioria dos organismos, os genes, assim como material genético dos organismos, são compostos de *DNA*. Existem alguns vírus, denominados retrovírus, que possuem material genético composto de *RNA* [52].

Em geral um gene está associado a um produto biológico expresso. Células diferentes e em estágios de desenvolvimento ou condições diferentes expressam genes distintos e em intensidades diversas [69].

Nos organismos eucariotos, os genes são geralmente compostos de partes chamadas *íntrons* e *éxons*, que se alternam dentro do gene. Na transcrição, os *íntrons* são retirados do *mRNA*. Assim, os *íntrons* correspondem a porções que não são utilizadas na síntese da proteína codificada pelo gene. A porção do *DNA* que corresponde a um gene completo é chamada *DNA* genômico, já uma porção que corresponde ao gene sem os *íntrons* é chamada de *DNA* complementar ou *complementary DNA* (*cDNA*). O *cDNA* pode ser obtido a partir do *mRNA* por meio do processo chamado transcrição reversa [69].

As moléculas de *DNA* são usualmente "empacotadas"em estruturas chamadas de **cromossomos**. A maioria das bactérias e vírus possuem um único cromos-

somo; os eucariotos usualmente possuem muitos. Um único cromossomo tipicamente contém vários genes [44]. O conjunto completo de cromossomos de uma célula é referido como **genoma**, isto é, o genoma pode ser definido como todo o conjunto de informações genéticas de um organismo, sendo constituído usualmente por uma ou mais moléculas de *DNA*, na vasta maioria dos seres, ou *RNA*, no caso de algumas famílias de vírus [64].

2.2 Seqüenciamento de Genomas

Seqüenciamento é a técnica utilizada para determinar a seqüência das bases que compõem o *DNA*, gerando seqüências que, num primeiro momento, não possuem nenhum significado biológico [74, 101].

O seqüenciamento é apenas a primeira etapa de um processo cujo resultado poderá apontar, por exemplo, novos métodos de diagnóstico, formulação de novos medicamentos, vacinas e novos tratamentos contra pragas, o que trará benefícios ao ambiente rural. Anualmente, são realizados em todo mundo milhares de projetos de seqüenciamento de genomas. A Figura 2.3 mostra gráficos com informações estatísticas sobre o quantitativo de projetos de seqüenciamento de genomas realizados em todo o mundo, de acordo com o Projeto *Genomes Online Databases* (*GOLD*) [108].

Nos projetos de seqüenciamento, existem dois possíveis caminhos para analisar o *DNA* de um organismo: realizar um seqüenciamento genômico estrutural ou funcional [123].

2.2.1 Seqüenciamento Estrutural

Os projetos estruturais possuem como finalidade seqüenciar todo o *DNA* genômico de um organismo, incluindo as partes que não codificam proteínas (*íntrons* e regiões intergênicas) [123].

De acordo com [120], uma análise detalhada de genomas estruturais predizem a presença de genes baseando-se na presença de seqüências como: códons de início e finalização; seqüências sinalizadoras de separação entre *exons* e a parte do *DNA* que expressará uma proteína. Entretanto, a presença de uma região codificadora não necessariamente implica que essa seqüência será transcrita e traduzida, na forma de uma proteína, com função bioquímica e biológica. A seqüência promotora de um gene, assim como a presença de outras seqüências reguladoras, é que irá regular quando, onde e como um gene será transcrito na forma de uma molécula de *mRNA* e, posteriormente, traduzido em uma proteína.

A seguir, descreveremos de forma resumida, dois projetos genoma estruturais, um envolvendo um consórcio internacional e outro no Brasil.

O seqüenciamento estrutural do genoma humano (Projeto Genoma Humano) foi formalmente iniciado em 1990, tendo durado treze anos. O término do projeto aconteceu em 2003, sendo que o seqüenciamento do genoma foi finalizado em abril de 2001. A coordenação deste projeto foi feita pelo *United States Departament of Energy (USDE)* e o *National Institutes of Health (NIH)* [150, 99]. Os principais objetivos do projeto foram:

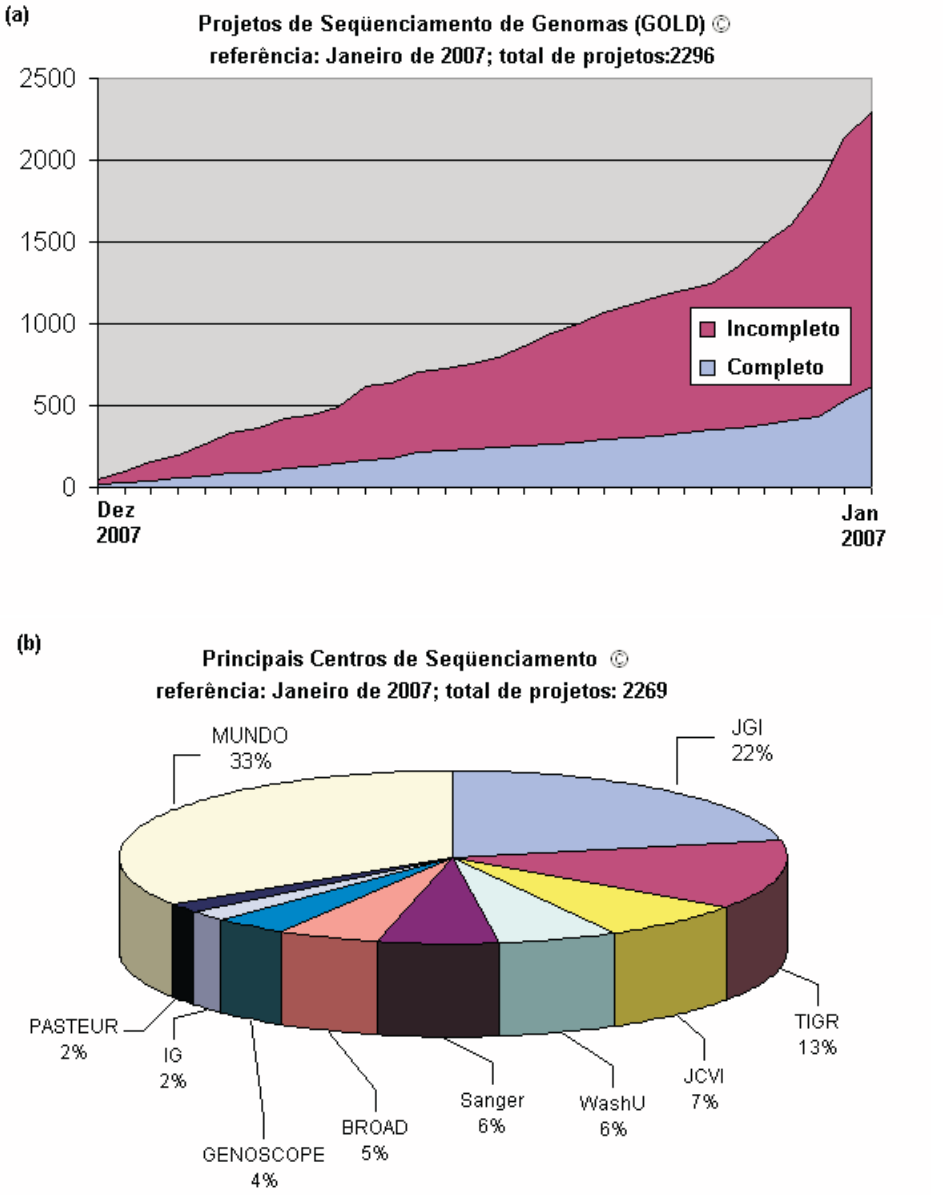


Figura 2.3: (a) Quantidade de projetos de seqüenciamento completos e incompletos. (b) Distribuição de projetos entre os principais centros de seqüenciamento do mundo (adaptados de [4]).

- Realizar a identificação completa dos aproximadamente 30.000 genes presentes no *DNA* humano;
- Determinar a seqüência dos cerca de 3 bilhões de bases que constituem o *DNA* humano;
- Armazenar todo o volume de informações em base de dados;
- Prover ferramentas computacionais para a análise dos dados;
- Permitir a transferência de tecnologias relatadas ao setor privado;
- Tratar das conseqüências éticas, legais e sociais que surgirem durante o desenvolvimento do projeto.

O Projeto Genoma Humano é de fundamental importância para a colaboração com a descoberta de várias doenças que atigem os seres humanos, como o câncer.

Na região Centro-Oeste temos vários projetos genoma em andamento. Citamos particulamente o Projeto Genoma Anaplasma, que consiste do seqüenciamento estrutural da bactéria (organismo procarioto) *Anaplasma marginale*. O foco é a doença anaplasmose bovina, que ocorre em áreas tropicais e subtropicais do mundo, acarretando em consideráveis prejuízos traduzidos por sintomas que refletem anemia, perda de peso, aborto e podendo levar o rebanho de gado a morte. O seqüenciamento e categorização funcional do genoma possibilitarão a descoberta acelerada de proteínas envolvidas em processos biológicos da riquétsia, potenciais alvos para o desenvolvimento de vacinas, para a ação de drogas e para utilização em imunodiagnóstico [10]. A análise computacional do projeto é realizada no Laboratório de Bioinformática da UnB. Os principais objetivos do projeto são:

- Seqüenciar o genoma de um isolado brasileiro de *Anaplasma marginale*;
- Realizar a anotação funcional das *ORF* identificadas no genoma;
- Avaliar a antigenicidade celular e humoral *in vitro* de proteínas recombinantes de *Anaplasma marginale*, previamente selecionadas segundo sua categorização funcional;
- Identificar antígenos promissores para imunização contra anaplasmose e imunodiagnóstico desta riquetsiose, bem como "proteínas-alvo"para a ação de drogas contra *Anaplasma marginale*.

2.2.2 *EST*

O *EST* foi um conceito criado no final dos anos 80, que constitui seqüências obtidas do mapeamento de uma porção do *DNA* complementar *(cDNA)*, gerado pela transcrição reversa do *mRNA* citoplasmático e pode ser feito pela extremidade $5'$ (*EST* $5'$) e/ou da extremidade $3'$ (*EST* $3'$) [53, 123].

Os *cDNAs*, quando seqüenciados, são chamados de *EST* (Figura 2.4). Os *EST* são seqüências obtidas de forma aleatória, geralmente incompleta, de uma porção

de *DNA* que representa um gene expresso e que pode ser utilizado para identificar genes do organismo que está sendo seqüenciado [101]. Genes expressos são genes cujo produto é uma proteína ou um *RNA* que está sendo produzido em um dado momento em uma célula. Geralmente, *EST* possuem tamanho pequeno variando em aproximadamente de 50 a 1000 bases e representam fragmentos de genes e não seqüências completas. Muitos centros de pesquisa que realizam seqüenciamento têm automatizado o processo de geração de *EST*, obtendo sua produção em taxas rápidas. Geralmente *EST* são seqüenciadas uma única vez.

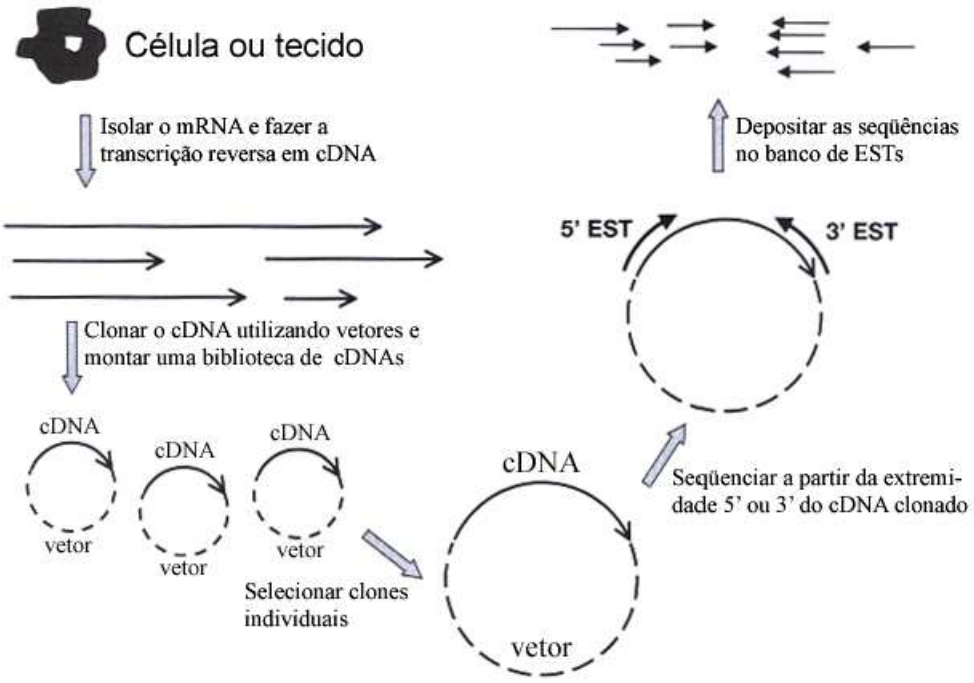


Figura 2.4: Visão geral de como um banco *EST* é construído [53, 123].

Desde a descrição original do projeto de seqüenciamento de 609 *EST*, em 1991, tem havido um acentuado crescimento no número de *EST* depositadas nos bancos públicos de seqüências [53]. Em meados de 1995, o número de *EST* no *GenBank* [88] ultrapassou o número de não-*EST*. Em Junho do ano 2000, os registros chegaram a 4,6 milhões de *EST*, constituindo 62% do total de seqüências depositadas no *GenBank*. Apesar do fato das *EST* originais serem de origem humana, o banco *dbEST* do *NCBI* contém aproximadamente 43.376.488 de *EST* (dados referentes a maio de 2007) de 1330 organismos [9].

De acordo com [101], considerando que os custos de projetos de seqüenciamento genômico completo (projetos estruturais) são elevados e que a quantidade de seqüências não codificadoras em organismos eucariotos são maiores que em organismos procariotos, os projetos de seqüenciamento genômico funcionais, com ênfase em *EST*, são mais comuns para eucariotos, enquanto que os completos são mais comuns para procariotos, quando o objetivo é obter somente as seqüências codificadoras. Para exemplificar, em organismos eucariotos, 90% do *DNA* não codifica para proteínas, tornando mais direta a obtenção das seqüências dos ge-

nes que as codificam. Porém, às vezes torna-se necessário o conhecimento de uma seqüência genômica completa, pois esta provê informações específicas as quais nem sempre podem ser obtidas apenas com o seqüenciamento de *EST* [123].

2.2.3 Seqüenciamento Funcional

Os projetos funcionais possuem enfoque no seqüenciamento de *EST* do organismo estudado, isto é, nas regiões gênicas cujos transcritos efetivamente são parte do *mRNA* final que será traduzido nos ribossomos [123, 96].

Em seguida, descreveremos dois projetos de seqüenciamento funcional realizados no Brasil.

O Projeto Genoma Pb tem como objetivo geral o mapeamento do genoma funcional e diferencial entre as formas de micélio e levedura do *Paracoccidioides brasiliensis*, um fungo (organismo eucarioto) dimórfico que causa a *Paracoccidioidomicose* (PCM), micose endêmica de alta prevalência na América Latina [82, 83].

O Projeto Genoma Pb baseou-se na identificação das *EST* de micélio e levedura dos genes de expressão diferencial, que potencialmente exerceriam funções relacionadas à adaptação do fungo ao hospedeiro, à manutenção do estado diferenciado, bem como com à virulência e/ou patogenecidade deste fungo, o qual sofre o processo de transição celular para infecção do hospedeiro humano [10]. Alguns dos principais objetivos do projeto foram:

- Identificação e seqüenciamento de 19.718 *EST* de micélio e levedura do fungo;
- Mapeamento do Genoma Funcional e Diferencial, por identificação das *EST* diferenciais, entre as suas formas de micélio e levedura;
- Realização da avaliação dos padrões expressos diferencialmente através da análise *Northern Blot* nas formas estágio-específicas e durante o dimorfismo, além de experimentos de *microarray* (microarranjo)[1];
- Caracterização e anotação funcional dos genes diferencialmente expressos.

O Projeto Genoma Guaraná será apresentado conforme [10]. Esse projeto está organizado dentro de uma rede de laboratórios na Região Amazônica (rede REALGENE), visando o seqüenciamento do genoma funcional do guaranazeiro. O guaranazeiro é uma das plantas (organismo eucarioto) nativas da Amazônia, de grande potencial econômico e está entre as mais estudadas tanto biológica como agronomicamente. A partir da realização desse projeto, os estudos realizados usando as técnicas modernas de genética genômica, promoverão o melhoramento do cultivo dessa planta, propiciando o desenvolvimento sustentável regional e benefícios nos âmbitos sociais, ambientais e econômicos.

O Projeto Genoma Guaraná tem um forte componente de formação de recursos humanos e criação de infra-estrutura tanto para seqüenciamento como para

[1]Em linhas gerais, microarranjo trata-se de uma tecnologia que permite testar genes diferenciais de milhares de genes simultaneamente.

a área de genética genômica, que permitirá a análise da diversidade genética de espécies da região amazônica tanto para orientar a conservação como para o uso sustentável dos recursos da biodiversidade.

O objetivo geral do projeto é estabelecer um programa de geração e análise de *EST* para diferentes órgãos e diferentes condições de cultivo e diferentes cultivares de guaranazeiro, clonando e seqüenciando as regiões ativas do genoma além de expandir o seqüenciamento, identificando regiões repetitivas não codificadoras que possam ser usadas como marcadores moleculares do tipo microssatélites.

As análises computacionais dos projetos genoma Pb e Guaraná foram realizadas no Laboratório de Bioinformática da UnB.

2.3 Comparação de Seqüências Genômicas

Do ponto de vista biológico, a comparação de seqüências é motivada pelo fato de que todos os organismos vivos são relacionados ao longo do processo evolucionário. Este fato implica que os genes ou proteínas (aminoácidos) de espécies próximas devem apresentar similaridades entre suas seqüências moleculares, indicando que devem possuir funções biológicas semelhantes [53, 68]. Em Bioinformática, a comparação de seqüências é uma das tarefas mais básicas a ser realizada e consiste em buscar partes semelhantes e diferentes entre seqüências.

A similaridade entre as seqüências é um valor que expressa quão semelhantes são duas seqüências. Para o cálculo da similaridade, pesquisadores utilizam um processo denominado alinhamento de seqüências.

O alinhamento de seqüências é um processo que sobrepõe as seqüências a serem analisadas afim de obter um nível de identidade entre elas. Este nível expressa a similaridade entre as seqüências comparadas. Um objetivo do processo de alinhamento de seqüências é permitir aos pesquisadores determinar a possibilidade de inferir a homologia entre duas seqüências [53, 68].

Um alinhamento de seqüências pode ser classificado como local ou global. No alinhamento local, apenas trechos (subcadeias) das seqüências comparadas são alinhados, sendo mais utilizado na busca por seqüências homólogas. No alinhamento global as seqüências são alinhadas em seu tamanho total, ou seja, uma seqüência p inteira alinhada com uma seqüência q inteira. O alinhamento global é comumente mais utilizado para determinar regiões "mais conservadas"(grande similaridade) de seqüências homólogas. Usualmente, para o cálculo da similaridade, *gaps* (espaços) nas seqüências são preenchidos com o símbolo "-". À máxima medida obtida no alinhamento de uma seqüência com as seqüências de uma base de dados, dá-se o nome de *score* (S) [9]. S indica o quão similar são duas seqüências, sendo que seu valor é obtido com o somatório do total de *matches* (símbolos coincidentes) e das penalidades aplicadas a *mismatches* (símbolos não coincidentes) e aos *gaps*. A fórmula que representa o valor de S é a seguinte:

$$S = max(\sum(matches, mismatches, gaps)) \tag{2.1}$$

Na Figura 2.5 podemos observar um exemplo de alinhamento global, sendo que o *score* é obtido de acordo com os seguintes valores: $+1$ para *matches*, -2 para *mismatches* e -1 para *gaps*.

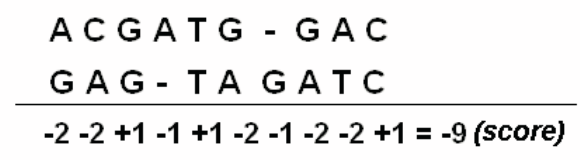


Figura 2.5: Exemplo de um alinhamento global.

O primeiro algoritmo utilizado para detecção de alinhamentos em seqüências é o de Needleman-Wunsch [119, 132] que utiliza a técnica de programação dinâmica (busca pelo alinhamento ótimo entre duas seqüências). Uma melhora desse algoritmo, apresentada em 1981, é o algoritmo de Smith-Waterman [139, 53], o qual permite que o alinhamento ótimo entre duas seqüências possa ser encontrado. Estes algoritmos requerem um tempo de execução proporcional ao tamanho das duas seqüências quem estão sendo comparadas (caso as entradas tenham tamanhos n e m respectivamente, a complexidade dos algoritmos fica na ordem de $O(nm)$). Como os tamanhos das entradas (tamanho das seqüências moleculares) para estes algoritmos constumam ser grandes, caso não exista recursos necessários para utilizá-los, a execução torna-se inviável (em relação a tempo de execução) [132, 101].

2.4 *Pipeline* de Bioinformática

Na Bioinformática, o termo *pipeline* corresponde a uma seqüência de processamento, onde os dados de saída (resultados) de uma etapa servem como dados de entrada para outra etapa [101, 69]. Os *pipelines* automatizam uma coleção de programas, que analisam dados experimentais e que ajudam um pesquisador a interpretar tais dados [101]. Tipicamente, os *pipelines* de seqüenciamento são executados em três grandes fases: submissão, montagem e anotação. A Figura 2.6 apresenta o diagrama de atividades relacionado as fases de um *pipeline* de seqüenciamento, onde o processo de anotação é constituído pelas atividades de anotação automática e manual. O diagrama está representado conforme a Linguagem de Modelagem Unificada ou *Unified Modeling Language* (*UML*) [12].

As fases de um *pipeline* possuem o objetivo de produzir seqüências de caracteres correspondentes aos fragmentos gerados nos laboratórios de Biologia Molecular, recompor trechos do *DNA* original e identificar funções e categorias nestes trechos de seqüências identificados [69]. A seguir descreveremos, de forma resumida, cada fase, bem como apresentaremos alguns exemplos de sistemas de *pipelines* existentes.

2.4.1 Submissão, Montagem e Anotação

Na fase de **submissão**, cada seqüência de uma placa é transformada em uma cadeia de caracteres (*string*) chamada *read* (*read* e seqüências serão tratados como sinônimos). Em uma *read*, para cada base da seqüência é associado um valor referente a probabilidade de erro na identificação da base. Normalmente, esta tarefa

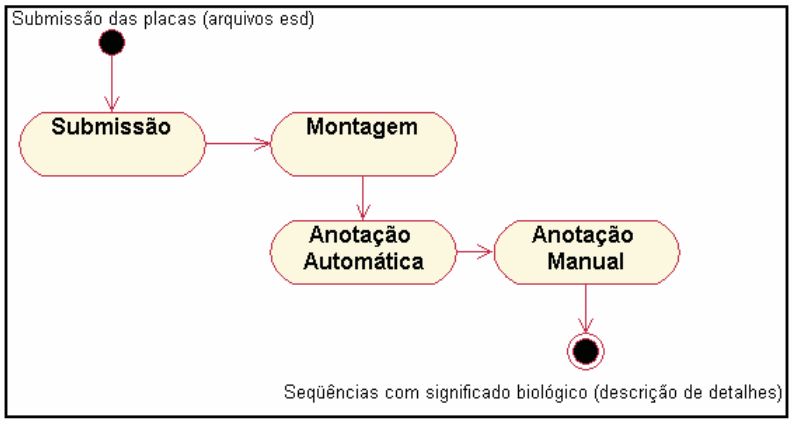


Figura 2.6: Principais fases de um *pipeline* de seqüenciamento.

é feita pelo programa *Phred*, que gera dois arquivos: um contendo uma cadeia composta por caracteres, correspondentes as bases, para cada *read* (arquivo de *reads*) e outro que contém a probabilidade de erro associada a cada base (arquivo de qualidade), ambos no formato *phd*. O programa *Phd2FASTA* é executado e converte ambos os arquivos *phd* em arquivos tipo texto no formato *FASTA* [69]. O formato *FASTA*, contém uma linha de cabeçalho (primeira linha), precedida pelo caractere "$>$", seguida por linhas que apresentam a seqüência. A primeira palavra do cabeçalho corresponde ao nome da seqüência, e o resto da linha corresponde a descrição da seqüência, contendo, respectivamente, o tipo de molécula, o nome do gene e o tamanho da seqüência. A Figura 2.7 apresenta um exemplo do formato *FASTA*. Neste exemplo, o nome da seqüência é *FOSB_MOUSE*, o tipo de molécula é *Protein* (Proteína), o nome do gene é *fosB* e o tamanho da seqüência é 338 bp. A unidade bp significa a quantidade de pares de bases ou *base pairs* encontrados na seqüência.

```
>FOSB_MOUSE Protein fosB. 338 bp
MFQAFPGDYDSGSRCSSSPSAESQYLSSVDSFGSPPTAAASQECAGLGEMPGSFVPTVTA
ITTSQDLQWLVQPTLISSMAQSQGQPLASQPPAVDPYDMPGTSYSTPGLSAYSTGGASGS
GGPSTSTTTSGPVSARPARARPRRPREETLTPEEEEKRRVRRERNKLAAAKCRNRRRELT
DRLQAETDQLEEEKAELESEIAELQKEKERLEFVLVAHKPGCKIPYEEGPGPGPLAEVRD
LPGSTSAKEDGFGWLLPPPPPPPLPFQSSRDAPPNLTASLFTHSEVQVLGDPFPVVSPSY
TSSFVLTCPEVSAFAGAQRTSGSEQPSDPLNSPSLLAL
```

Figura 2.7: O formato *FASTA* [5].

Cada seqüência é filtrada para remover porções que provavelmente não pertencem ao organismo sendo estudado, mas que pertencem a vetores (seqüências dos organismos usados para replicar o *DNA* do organismo que está sendo estudado) e contaminantes (seqüências de *DNA* de outros organismos). As *reads* são filtradas utilizando-se programas tais como o *Crossmatch*. Os trechos identificados como

sendo de outros organismo são mascarados com o caractere X [69].

Podem existir casos onde possam ser feitas análises de redundância entre as seqüências de uma placa. Para tanto, pode-se executar um programa de montagem de seqüências, como o *CAP3*. Os agrupamentos gerados pelo *CAP3* indicam seqüências muito similares e provavelmente redundantes dentro da placa submetida [69].

A fase de **montagem** consiste em gerar agrupamentos de seqüências similares, isto é, seqüências que têm prefixos e sufixos "aproximadamente iguais"[69]. Duas seqüências são semelhantes se há similaridades entre o sufixo de uma e o prefixo de outra. Esta correspondência é conhecida como alinhamento. Estes agrupamentos buscam unir fragmentos que potencialmente pertencem à mesma região do *DNA*. Grupos formados por mais de uma seqüência são chamados *contigs* e grupos formados por uma única seqüência são chamados *singlets*. Para cada *contig*, uma seqüência consenso é gerada e esta representa o *contig*.

Para a montagem das seqüências, normalmente são utilizados os programas *Phrap* e *CAP3*. Ambos geram arquivos contendo dados sobre a montagem, os alinhamentos e as qualidades das seqüências que compõem um *contig*, e um outro que contém as seqüências dos *singlets* em formato *FASTA*. Apesar de outros arquivos também serem gerados, apenas os arquivos de *contigs* e *singlets* são utilizados nos processamentos seguintes [69].

A identificação de possíveis genes nas seqüências (seqüências *contigs* ou *singlets*) é feita através de programas como o *Glimmer*, que identifica posições iniciais ou finais de uma região que possivelmente identifique um gene. No caso dos projetos estruturais, deve-se identificar os genes dentro do *DNA*. Como o *Glimmer* pode gerar resultados falsos positivos e falsos negativos, os biólogos analisam outras informações para identificar um gene, como por exemplo a presença de *tRNA* (um outro tipo de gene, não detectado pelo *Glimmer*), terminadores e sítios de ligação ribossomal. Neste caso, programas como *tRNAScan-SE* (busca de *tRNA*) [111, 110], *Transterm* (busca de terminadores) [13] e *RBSFinder* (busca sítios de ligação ribossomal) [142] são exemplos de *softwares* que podem vir a serem utilizados [101]. As posições de cada "candidato"a gene (*ORF*) são armazenadas em um banco de dados, juntamente com as seqüências *contigs* e os *singlets*.

Nos projetos de seqüenciamento de *EST*, não é necessário identificar genes, uma vez que as *EST* representam porções do *DNA* que são expressas (já correspondem, portanto, a genes). Neste caso, apenas as seqüências *contigs* e os *singlets* são armazenados diretamente no banco de dados [69].

Por sua vez, a fase de **anotação** tem por objetivo agregar significado biológico às seqüências geradas na fase de montagem. A anotação genômica é o processo de interpretar os dados do seqüenciamento de genomas, tornando-os informações biologicamente relevantes, ou seja, saber o que representa cada seqüência encontrada no seqüenciamento [90]. Embora a anotação de um genoma completo, conhecida por anotação estrutural, forneça uma perspectiva geral ela não descreve detalhadamente todos os genes, sendo que está descrição detalhada consiste da anotação funcional.

A tarefa de determinar funções de genes e proteínas nos organismos, também conhecida como anotação funcional, foi e continua sendo uma das principais tarefas realizadas em Bioinformática. Nos últimos anos, muitas metodologias compu-

tacionais foram desenvolvidas para auxíliar a solução desta tarefa pelos pesquisadores. Para um melhor entendimento, o desenvolvimento destas metodologias computacionais é abordado em três níveis, levando em consideração a quantidade e o tipo de informação utilizada [104]. No primeiro nível, estão métodos para comparação e busca de similaridade entre pares de seqüências (*BLAST* [14] e *FASTA* [15], são exemplos de *softwares* utilizados); no segundo nível, estão métodos utilizados para lidar com múltiplas seqüências, com objetivos de, por exemplo, coletar estatísticas para prover maior acurácia às anotações (por exemplo, o método *profiles Hidden Markov Models* (perfis *HMM*) [126, 98]); finalmente, no terceiro nível, encontram-se métodos que buscam informações além das fornecidas através de busca de similaridade. Em [94], temos alguns exemplos de métodos utilizados neste nível, como *DNA Microarray gene expression data* (Expressando dados de genes em Microarranjos de *DNA*), *phylogenetic profiles* (perfis filogenéticos) e *genetic networks* (redes genéticas).

A anotação de genomas é geralmente dividida em dois passos. Primeiro, a anotação automática em que devem ser comparadas todas as seqüências do projeto com seqüências de bancos de dados públicos e/ou privados. As funções e categorias das seqüências estudadas são inferidas por comparações com seqüências semelhantes que tiveram suas funções e categorias previamente determinadas. O segundo passo, a anotação manual, à qual este trabalho está relacionado, é feita pelos biólogos que utilizam as informações da anotação automática, bem como seus conhecimentos para decidir, por exemplo, genes, os produtos e suas funções associada à seqüência.

Os passos básicos envolvidos em uma anotação estrutural ou funcional incluem [53]:

- uma busca de seqüências similares ao do organismo estudado;
- identificar motivos ou *motifs* e domínios estruturais por comparação de seqüências protéicas, a partir de bancos de dados, como por exemplo, o *Pfam* [51] e o *Simple Modular Architecture Research Tool* (*SMART*); [133];
- predizer características de estruturas protéicas;
- gerar predições acerca de estruturas secundárias e, se possível, terciárias das proteínas.

2.4.2 Exemplos de Anotação Automática

Como descrito anteriormente (Seção 2.4.1), a anotação automática usa *softwares* e bases de dados disponíveis na *web*, a fim de inferir funcionalidades de seqüências estudadas a partir de anotações já conhecidas das seqüências das bases de dados públicas. A seguir, apresentaremos de forma esquemática três processos de anotação automática dos projetos genoma Pb, Guaraná e Anaplasma. Alguns *softwares* relacionados, bem como as bases de dados, estão descritos, respectivamente, nas Seções 2.5 e 2.6.

No Projeto Genoma *Pb os softwares* utilizados para a anotação automática foram o *BLASTX* [14], *FASTA*, *InterProScan* [5] e *Prediction of Protein Sorting*

Signals and Localization Sites In Amino Acid Sequences (*PSORT*) [117], e as bases foram as seguintes: *non-redundant* (*nr*) [9]; *xyva - Clusters of Orthologous Groups of proteins* (*COG*) [147]; bases dos organismos (fungos) *Saccharomyces cereviseae* [16] e *Schizosaccharomyces pombee* [17]; projeto *Gene Ontology* (*GO*) [47]; base *Interpro* (base de famílias protéicas) [5]. A Figura 2.8 exibe o processo de anotação automática do Projeto Genoma Pb.

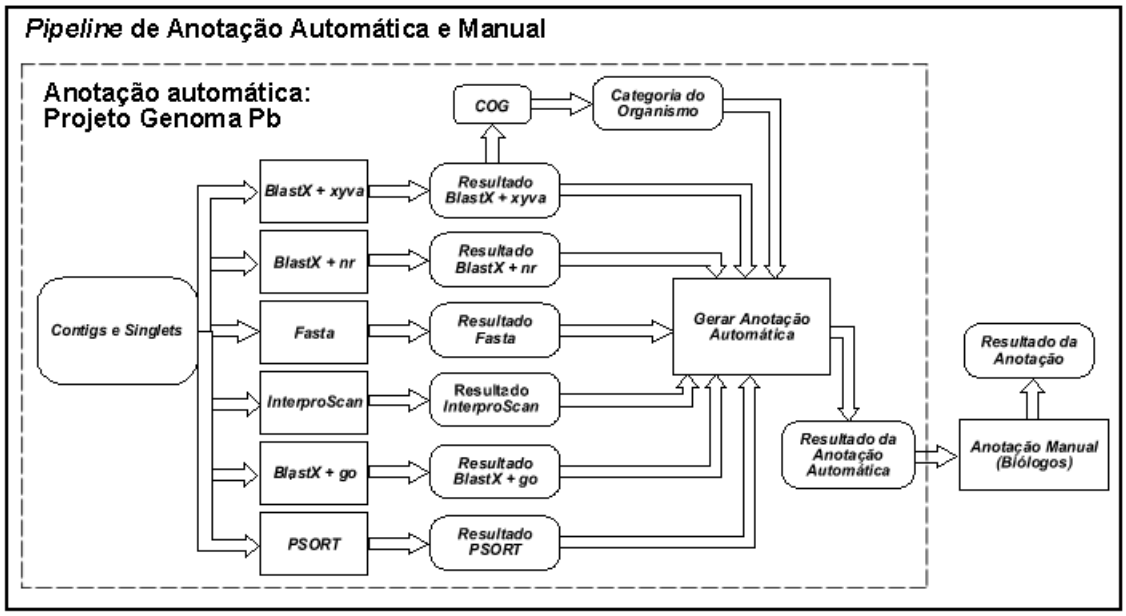


Figura 2.8: A anotação automática do Projeto Genoma Pb (adaptada de [123]).

No Projeto Genoma Guaraná o *software* utilizado foi o *BLASTX*, aplicados às bases *nr* [9], *EuKaryotic Orthologous Groups* (*KOG*) [146], projeto *GO* [47] e *Swiss-Prot* [18]. Foram identificados domínios de famílias proteícas, sendo utilizado para este propósito a base *Pfam* [6]. A Figura 2.9 exibe o processo de anotação automática do Projeto Genoma Guaraná.

O Projeto Genoma Anaplasma, diferente dos projetos genoma Pb e Guaraná, utilizou o *software BLASTP* [14] do pacote de ferramentas *BLAST*. Os outros três sistemas utilizados pelo projeto foram o *InterProScan*, *PSORT* e o *tRNAscan-SE* [111, 110]. Em relação as bases de dados, as utilizadas pelo *BLASTP* foram *cog*, *nr*, *Swiss-Prot*. As bases comumente utilizadas pelo *InterProScan*, *PSORT* e pelo *tRNAscan-SE* são, respectivamente, *Interpro*, *PSORTdb*, *Sprinzl tRNA*, *GenBank tRNA* e a *The Genomic tRNA database* (*GtRNAdb*). A Figura 2.10 exibe o processo de anotação automática do Projeto Genoma Anaplasma.

2.4.3 Sistemas de *Pipeline*

Nessa seção descreveremos alguns sistemas de *pipeline* para projetos de seqüenciamento. A maioria destes trabalhos está disponível para uso em pesquisas mediante autorização dos autores. Caso o uso seja comercial, há a disponibilidade

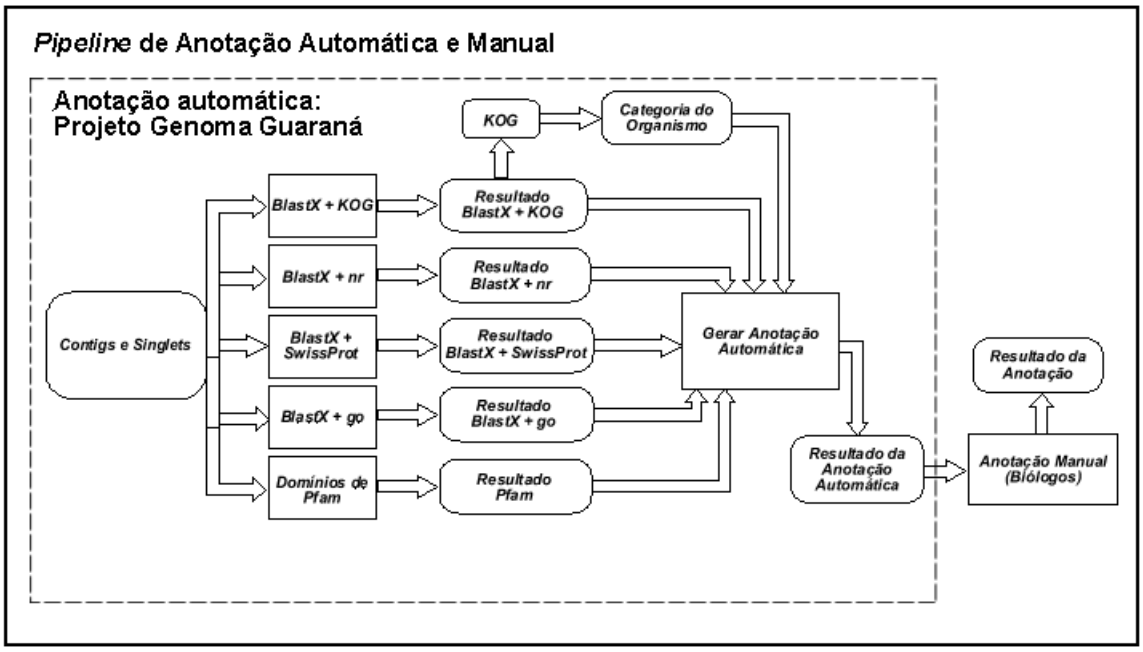


Figura 2.9: A anotação automática do Projeto Genoma Guaraná.

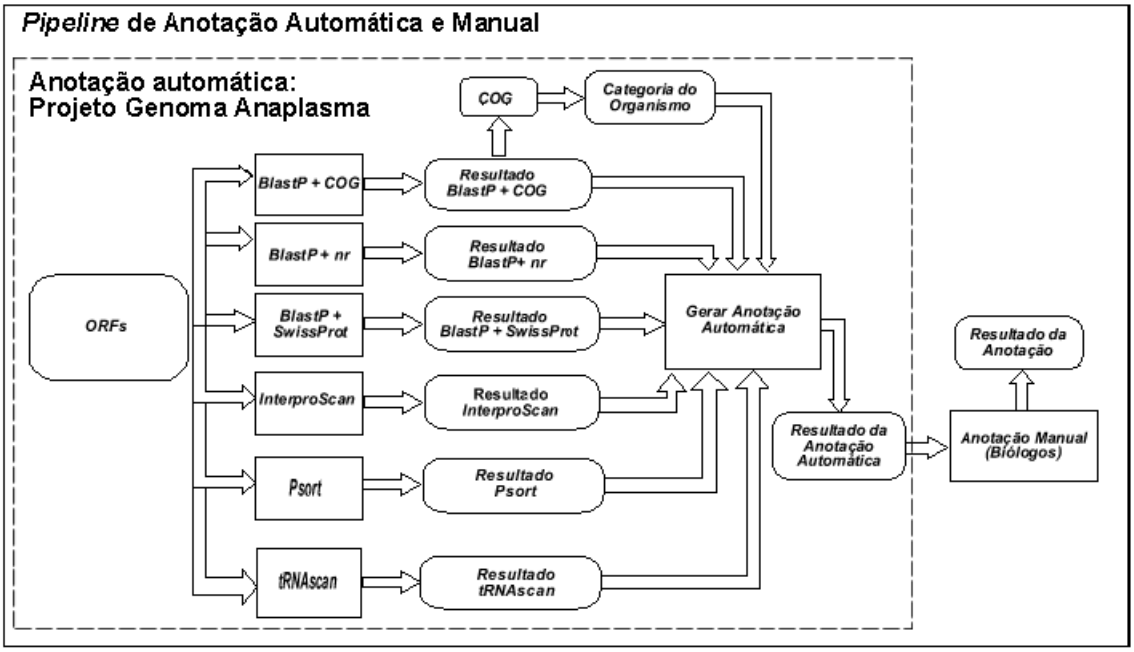


Figura 2.10: A anotação automática do Projeto Genoma Anaplasma.

de obter licenças específicas. Nestes projetos, a anotação manual é de única responsabilidade do especialista humano (anotador), não estando disponibilizadas, em geral, *software* de apoio que sugira anotações aos anotadores.

A ferramenta *Genomic Analysis Resources for a Sequence Annotation* (*GARSA*) é um sistema *web* e foi desenvolvido para analisar dados genômicos, usando um *pipeline* composto por alguns pacotes de *softwares* para Bioinformática [77]. Na versão atual da ferramenta, o sistema permite analisar *EST*, *Orfs EST sequences* (*Orestes*) e dados *Genome Survey Sequence (GSS)*, aceitando como entradas cromatogramas (arquivos onde cada base da seqüência está representada por um pigmento colorido), *downloads* de seqüências do *GenBank*, arquivos *FASTA* armazenados localmente ou a combinação de todas essas entradas.

A ferramenta *BioNotes* é um sistema para anotação de bioseqüências com o objetivo de auxiliar os pesquisadores a buscar bases de dados externas, executar programas de análise, analisar anotações e adicionar novas anotações para registrar interpretações dos dados [101]. O sistema *BioNotes* possui dois tipos básicos de *pipelines* definidos: um para projetos de seqüenciamento de genomas completo e outro para projetos de seqüenciamento de genomas de *EST*, isto é, projetos de seqüenciamento de genomas funcionais. O *BioNotes* foi desenvolvido para ambiente *web*, utilizando uma arquitetura cliente servidor, permitindo aos usuários o compartilhamento de suas anotações na *web*. Para ilustrar e oferecer um melhor entendimento, a Figura 2.11 mostra como funciona o *pipeline* para projetos de seqüenciamento de genomas de *EST*.

O *System for Automated Bacterial Integrated Annotation* (*SABIA*), é um *pipeline* para seqüenciamento de genomas de procariotos, considerado um *software* de referência desenvolvido no país [42]. A ferramenta *SABIA* foi desenvolvida, para ambiente *web*, pela equipe do Laboratório Nacional de Computação Científica do Ministério da Ciência e Tecnologia (LNCC) e pela Universidade Federal do Rio de Janeiro (UFRJ) e realiza tarefas automáticas de montagem, análise e identificação de *ORF* e análise de regiões extragênicas. O *SABIA* compreende um conjunto de *scripts Perl/CGI* que manipulam os dados armazenados em uma banco de dados relacional (*MySQL*). Os *scripts* geram relatórios *HTML* e formulários que permitem aos usuários acessarem informações biológicas e carregar os resultados de suas análises no banco de dados [42]. Os módulos de anotação e montagem podem ser executados pelos usuários de forma independente. O *software* foi desenvolvido para ser executado nas plataformas *Linux* e *Solaris* e é distribuído sob licença proprietária. Executa rotinas diárias de monitoramento de execução de seqüenciamentos, montagem e atualização de cópias locais de bancos de dados públicos [42].

O *software GenDB* fornece suporte a estratégias de anotação manual bem como anotação automática. O sistema é do tipo *pipeline* e foi desenvolvido para prover a anotação de genomas procariotos [115]. O *GenDB* foi utilizado em dezenas de projetos de anotação de micróbios, sendo também utilizado como um *framework* para avaliação, em larga escala, de diferentes estratégias de anotação. A funcionalidade central do sistema é oferecer aos pesquisadores funções simples para prover anotação automática. O *GenDB* é *open source*, está sob licença *GNU General Public License* (*GPL*) e no caso de uso comercial existem licenças específicas.

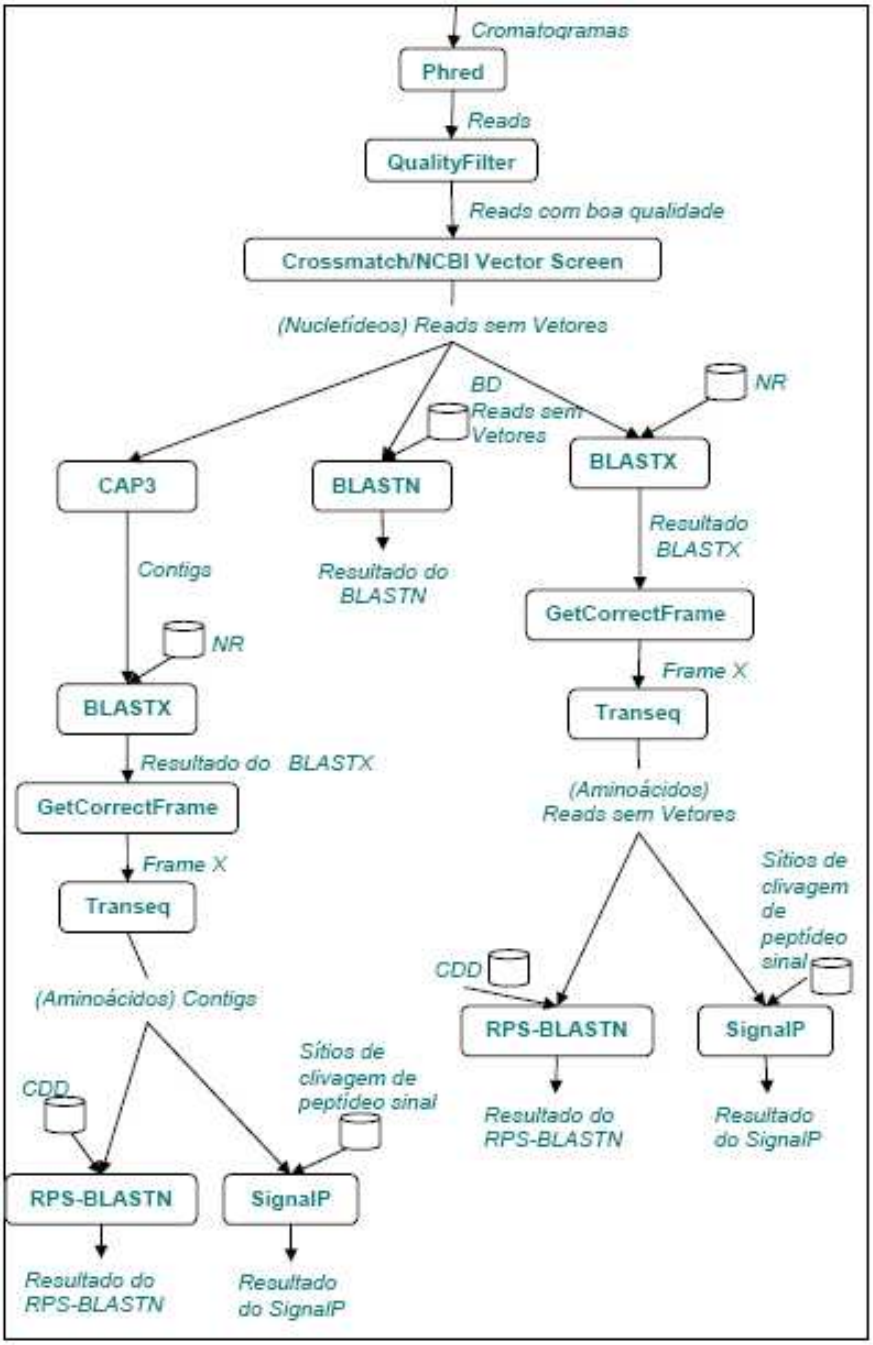


Figura 2.11: Figura referente ao *pipeline* para projetos genoma de *EST* no sistema *BioNotes* [101].

O sistema Timina [69] é um *framework* desenvolvido no Laboratório de Bioinformática da UnB, capaz de suportar as fases de submissão, montagem e anotação em projetos de seqüenciamento de genomas. O *framework* foi desenvolvido para ambiente *web*. Atualmente o Timina dá suporte ao *pipeline* do Projeto Genoma Anaplasma. O *framework* Timina foi completamente implemetado em *software* livre e está disponibilizado sob licença *GPL*.

2.5 Ferramentas para Anotação

Nessa seção são apresentados alguns *softwares* freqüentemente utilizados em projetos de Bioinformática.

FASTA

A ferramenta *FASTA* possui método heurístico para realizar comparação de *strings* [109, 124]. Incialmente, o algoritmo de Wilbur-Lipman [155] foi proposto para analisar similaridade entre seqüências de proteínas e entre seqüências de *DNA*. Em 1985, [109] criaram a ferramenta *FASTP*, que pesquisa bases de dados de aminoácidos utilizando uma técnica de busca de identidades compartilhadas entre duas seqüências e explorando restrições biológicas relacionadas à evolução molecular. Em 1988, [124] aperfeiçoaram a ferramenta *FASTP* criando a ferramenta *FASTA*, cujo algoritmo aumenta a sensitividade com pequena perda de seletividade [140].

Na Bioinformática, a sensitividade é medida pela taxa de falsos negativos. Falso negativo é um resultado, por exemplo, um alinhamento que o programa de comparação deixa de reportar, apesar dele ser biologicamente verdadeiro. Quanto menor a fração de falsos negativos, maior é a sensitividade de um programa. Em relação a seletividade, esta é medida pela taxa de falsos positivos. Falso positivo pode ser, por exemplo, um alinhamento retornado por um programa mas que não corresponde a um resultado biologicamente verdadeiro (pode ter sido um acaso). Quanto maior a taxa de falsos positivos, menor é a seletividade de um programa [135].

FASTA pode ser vista como uma combinação do programa *FASTP* (proteínas) com *FASTN*, sendo este último um programa projetado especificamente para analisar seqüências de nucleotídeos [123].

Uma melhoria apresentada por *FASTA*, em relação ao *FASTP*, relaciona-se com a computação das pontuações iniciais. O programa *FASTA* executa um passo adicional depois que as melhores regiões foram selecionadas: tenta juntar seqüências que estejam em regiões vizinhas, mesmo que elas não pertençam à mesma diagonal na matriz de substituição (*PAM* ou *BLOSUM*) [53, 132, 123]. Com isso, as pontuações iniciais melhoram signicativamente para as seqüências relacionadas. Além disso, as dez melhores regiões são escolhidas em *FASTA*, enquanto que no *FASTP* eram apenas cinco.

A partir da versão 2.0, o programa *FASTA* provê uma estimativa estatística significativa para cada alinhamento encontrado, que é a possibilidade de fazer embaralhamentos locais nas seqüências [15]. Observou-se que freqüentemente um

alinhamento não significativo biologicamente apresentava uma pontuação alta. O embaralhamento local amenizou esse problema ajudando a detectar os alinhamentos aleatórios. Outras melhorias no *FASTA* incluem a flexibilidade de escolha da matriz de substituição e a possibilidade de computar mais de uma pontuação para cada seqüência embaralhada [132].

BLAST

A família de ferramentas *BLAST* [43], desenvolvida pelo *NCBI*, está entre os *softwares* para análise de seqüências nucléicas, ou proteícas, mais utilizados no mundo [114]. Sua função é realizar análises de similaridades entre duas seqüências em bancos de *DNA* ou proteínas, com base em um algoritmo heurístico, podendo ser utilizado através de uma plataforma distribuída ou como uma aplicação *stand alone*[2]. O *BLAST* pode também ser utilizado para inferir relacionamentos funcionais e evolucionários entre duas seqüências bem como auxiliar a identificar membros de famílias de genes.

A primeira versão foi desenvolvida em 1990 e tinha por objetivo pesquisar similaridades locais entre seqüências sem levar em consideração os *gaps*. A motivação para seu desenvolvimento foi a necessidade de melhorar o desempenho do algoritmo utilizado na ferramenta *FASTA*, por meio de busca de k-tuplas (subseqüências idênticas) em menor número e de maior valor qualitativo. A idéia era integrar o uso de matrizes de substituição na primeira etapa do algoritmo, onde eram buscados as k-tuplas [140].

Ao realizar uma análise, a ferramenta *BLAST* retorna uma lista de alinhamentos - lista de *hits*[3] - ou seja, fragmentos de uma sequência, com o *score* entre a seqüência que está em análise e as seqüências contidas no banco de dados pesquisados.

No que refere-se ao funcionamento da ferramenta, dadas duas seqüências quaisquer (observando que o *BLAST* pode fazer pesquisas em diversas bases de seqüências), são computados todos os possíveis pares de segmentos cujos *scores* estejam acima de um certo limite que pode ser definido pelo usuário. Ao par que possui o *score* máximo é dado o nome de Par de Segmento Máximo ou *High-Scoring Segment Pair* (*HSP*), e é este *score* que indica a similaridade entre as duas seqüências [14].

O *BLAST*, ao realizar uma busca por seqüências similares à passada como parâmetro, fornece algumas informações estatísticas, sendo que as informações mais importantes são o *score*, *expectation value* (*e-value*) e o *probability-value* (*p-value*). O *score* é a medida correspondente ao quanto duas seqüências são similares. O *e-value* corresponde a possibilidade de um determinado alinhamento acontecer por acaso. Quanto menor o *e-value*, menor a chance de que o alinhamento tenha acontecido por acaso, e portanto, maior a chance do alinhamento evidenciar uma similaridade real (maior significância) [135]. A relação entre o *e-value* e o *score* é inversamente proporcional, isto é, quanto mais significante for o *score* menor será o *e-value*. O *p-value* é a probabilidade de um alinhamento *HSP*

[2]Aplicações isoladas, *desktop*

[3]São seqüências coincidentes ou similares àquela informada como parâmetro de busca [65].

ocorrer com valor maior ou igual a um determinado *score*. Sua interpretação é semelhante a do *e-value* (quando o *e-value* < 0.00001, *p-value* e *e-value* são aproximadamente idênticos). A relação do *p-value* com o *e-value* é determinada pela seguinte fórmula [14]:

$$P - value = 1 - e^{-(E-value)} \tag{2.2}$$

O conjunto de programas do *BLAST* compreende as ferramentas *BLASTN*, *BLASTP*, *BLASTX*, *TBLASTN* e *TBLASTX*. A Tabela 2.1 descreve resumidamente os objetivos de cada ferramenta.

Tabela 2.1: Conjunto de programas *BLAST*.

Programa	**Base de Dados**	**Consulta (*Query*)**
BLASTN	Nucleotídeo	Nucleotídeo
BLASTP	Proteína	Proteína
BLASTX	Proteína	Nucleotídeo transformado em proteína
TBLASTN	Nucleotídeo transformado em proteína	Proteína
TBLASTX	Nucleotídeo transformado em proteína	Nucleotídeo transformado em proteína

PSORT

O *PSORT* é um pacote de programas, desenvolvido na Universidade de Tokyo em 1991, utilizado para a predição da localização de proteínas nas células [117]. De acordo com o tipo de organismo ao qual a proteína pertence, um conjunto de localizações possíveis é testado.

No caso das bactérias Gram-negativas[4], *PSORT* verifica se a proteína pertence ao citoplasma, à membrana interna, à membrana externa ou ao periplasma da célula[5]. Após a análise dos dados (por exemplo, seqüência de aminoácidos e seqüências da bactéria Gram-negativa), o programa fornece, no arquivo de saída, uma predição da localização da proteína e a sua probabilidade de ocorrência. Originalmente, o *PSORT* foi desenvolvido para predição de proteínas em bactérias Gram-negativas, mas o pacote de ferramentas *PSORT, PSORT II, iPSORT, PSORT-B e WoLF PSORT* já foi expandido para realizar a predição em diversas classes de organismos [19].

HMMER

[4]Bactérias Gram-negativas são bactérias que não retêem a tintura cristal violeta de acordo com o protocolo de classificação Gram. As bactérias Gram-negativas possuem uma parede celular dupla, em que a interna é uma fina camada de peptidoglicanos (responsável pela forma das células e proteção do citoplasma; confere rigidez ao corpo bacteriano), enquanto que a exterior á formada por carboidratos, fosfolípidos e proteínas [`http://pt.wikipedia.org/wiki/Bactria,http://www.forp.usp.br/restauradora/calcio/citolog.htm`]

[5]Corresponde a um espaço situado entre a membrana externa e membrana citoplasmática das células Gram-negativas.

O *HMMER* é um *software* que implementa perfis *HMM* para análise de seqüências biológicas. *Hidden Markov Models* (*HMM*) são modelos baseados em cálculos probabilísticos [98, 76, 80]. Uma das vantagens de utilizar *HMM* para a construção de perfis é que eles são considerados um bom método para tratar com *gaps* encontrados em famílias protéicas [138].

O *software HMMER* é utilizado para buscas "sensíveis"em bases de dados usando alinhamento múltiplo de seqüências. O usuário fornece como entrada para o programa um alinhamento múltiplo de seqüências. Então um *HMM* é construído afim de que o usuário possa utilizá-lo como uma consulta dentro da base de dados de seqüências para encontrar (e/ou alinhar) homologias adicionais de uma família protéica. O *software HMMER* é comumente utilizado com a base *Pfam*, que está descrita posteriormente. O *HMMER* é disponibilizado sob licença pública *GPL* [20].

INFERNAL

O *INFERNAL* é um pacote de programas que permite ao usuário criar perfis de estruturas secundárias[6] de *RNA*, e utilizá-los para procurar por *RNAs* homólogos em bases de seqüências de ácidos nucléicos, ou criar novas estruturas baseadas em alinhamentos múltiplos de seqüências [79].

O pacote de ferramentas *INFERNAL* é o resultado da implementação de modelos probalísticos para representar perfis de estruturas secundárias de *RNA*, os quais são chamados de Modelos de Covariância (MC) [78]. Os MC são análogos aos perfis obtidos por *HMM*, utilizados em análise de seqüências protéicas, sendo utilizados para inferir segmentos não codificadores na estrutura de *RNA* não-codificadores (*ncRNA*)[7]. Este fato permite o *INFERNAL* ser comparável à ferramenta *HMMER*. Entretanto, os MC incluem informações de consenso das estruturas secundárias de *RNA* e da seqüência.

De acordo com [78], a complexidade de um algoritmo de programação dinâmica para alinhamento de um MC com uma seqüência de *RNA* de tamanho N é $O(N^3)$, tornando-o prático apenas para tamanhos pequenos de seqüências de *RNA*. O algoritmo utilizado pelo *INFERNAL* é uma variante, utilizando como idéia central a estratégia "dividir para conquistar", análoga aos eficientes algoritmos de alinhamento linear de seqüências de Myers/Miller [116] e possui uma complexidade na ordem de $O(N^2 \log N)$, o que torna possível obter alinhamentos ótimos de estruturas em seqüências de *RNA*.

O pacote *INFERNAL* disponível para uso contém programas, dentre outros, que permitem que os usuários criem MC de estruturas secundárias de *RNA*, alinhem novas seqüências com um MC e busquem seqüências homólogas de *RNA* para um MC [20]. A base de dados freqüentemente utilizada em conjunto com o *INFERNAL* é a base *Rfam* [91], a qual descreveremos na Seção 2.6.

[6]Estruturas espaciais do *RNA*, i.e, a forma apresentada pela estrutura de *RNA*. Alguns exemplos são as formas de "grampo"e "dupla-fita"

[7]RNA não codificadores são seqüências que foram transcritas mas não participaram da síntese protéica, ou seja, não foram traduzidas em proteínas [`http://en.wikipedia.org/wiki/Non-coding_RNA`].

O *INFERNAL* está passando por uma fase exaustiva de testes. O *software* - observando a última versão da ferramenta (versão 0.72), disponibilizada em janeiro de 2007 - gera saídas robustas, porém com falta de dados ou *missing values* acerca de características relevantes aos especialistas. É importante ressaltar que os algoritmos base do *INFERNAL* requerem um enorme tempo de processamento. Portanto, esta ferramenta é lenta e os projetos que venham a utilizá-la provavelmente vão requerer o uso de vários processadores [79].

tRNAscan-SE

O *tRNAscan-SE* é um *software* utilizado para identificar *tRNA* em seqüências de *DNA* [20]. A ferramenta possui uma probabilidade de acerto na identificação de *tRNA* na ordem de 99-100%, com uma taxa de falsos positivos menor que 1 por 15 gigabases (10^9 bases). Em conjunto com o *tRNAscan-SE*, são utilizados dois outros programas para detecção de *tRNA*, são eles: *EufindtRNA* e *COVE*. Em uma primeira etapa, os programas *tRNAscan-SE* e *EufindtRNA* identificam, nas seqüências, regiões "candidatas"a *tRNA*. Estas regiões (subseqüências) são então passadas para a análise da ferramenta *COVE*, visando a confirmação da predição inicial de *tRNA*. Em relação ao tempo de busca nas bases de *tRNA*, a ferramenta busca aproximadamente 30.000 bp por segundo [111, 110].

2.6 Bancos de Dados para Anotação

Com os diversos projetos de seqüenciamento espalhados pelo mundo, o crescimento das bases de dados biólogicos, principalmente as bases públicas, têm apresentado crescimentos exponenciais em seus tamanhos. Nessa seção serão apresentadas algumas das principais bases de seqüências utilizadas por pesquisadores em todo o mundo.

GenBank/EMBL/DDBJ

O *GenBank* [9] é a principal base de dados do consórcio *International Nucleotide Sequence Database Collaboration (INSDC)*, o qual é formado pelas bases *GenBank* (*USA*), *European Molecular Biology Laboratory-EMBL* (Europa) e *DNA DataBank of Japan-DDBJ* (Japão). O *GenBank* contribuiu muito para a marca de mais de 100 gigabases de seqüências de *DNA* disponibilizadas pelo *INSDC* [57]. O *INSDC* fornece acesso livre e irrestrito aos dados, bem como mantém permanentemente registros das bases. A Figura 2.12 ilustra a dimensão das bases de dados *GenBank*, *EMBL*, *DDBJ* e a interação entre elas.

O *GenBank*, como visualizado na Figura 2.13, vêm apresentado nos últimos anos um comportamento exponencial em seu crescimento. Mais de 140.000 espécies estão representadas e novas espécies estão sendo adicionadas a uma taxa de mais de 1.700 por ano. Depois do *Homo sapiens*, as principais espécies no *GenBank* em termos de número de bases são *Mus musculus*, *Rattus norvegicus*, *Bos taurus*, *Danio rerio* e *Zea mays* (dados relativos ao *release* 158 - fevereiro de 2007).

Cada entrada do *GenBank* inclui uma descrição concisa da seqüência, o nome

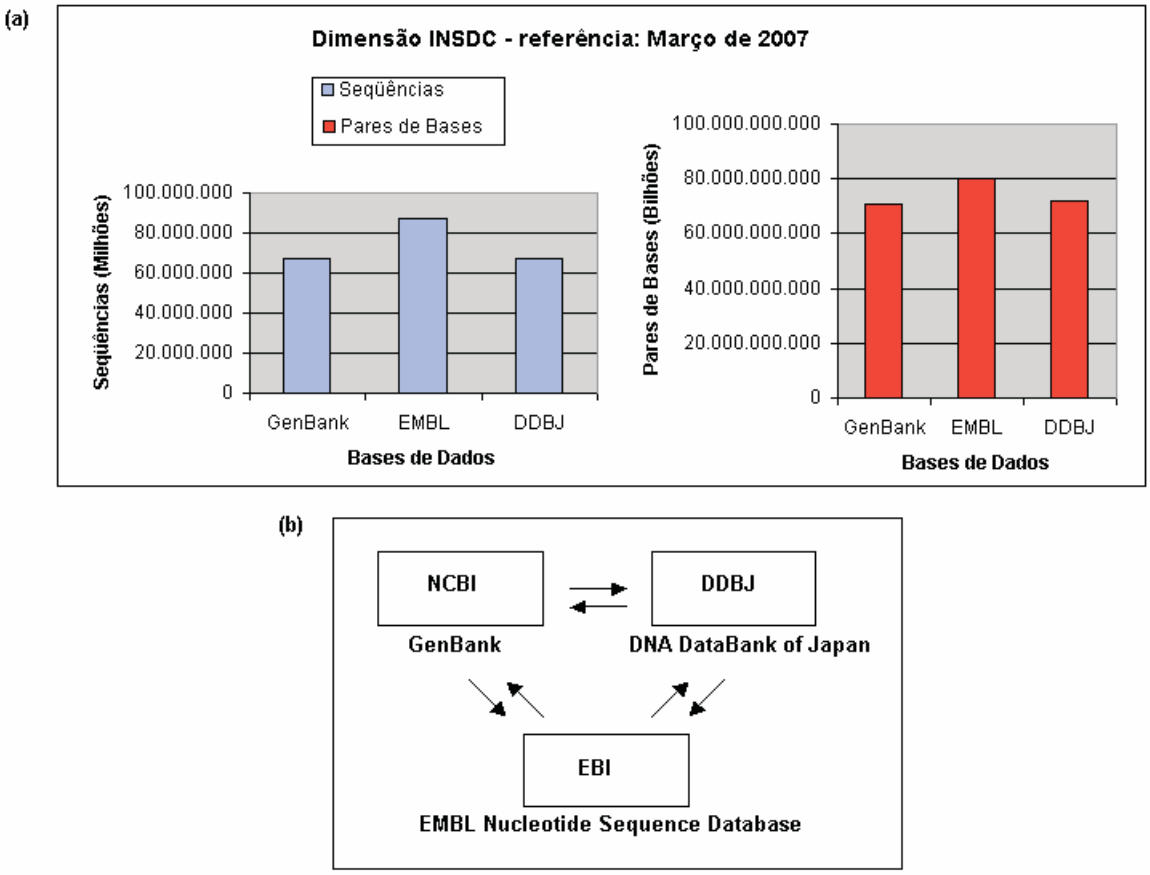


Figura 2.12: (a) Dimensões, em número de seqüências e pares de bases, das bases de dados, de acordo com os útimos *releases* disponibilizados: *GenBank* (*release* 158 - fevereiro de 2007), *EMBL* (*release* 90 - março de 2007) e *DDBJ* (*release* 69 - março de 2007). (b) Interação entre as bases de dados do consórcio *INSDC*.

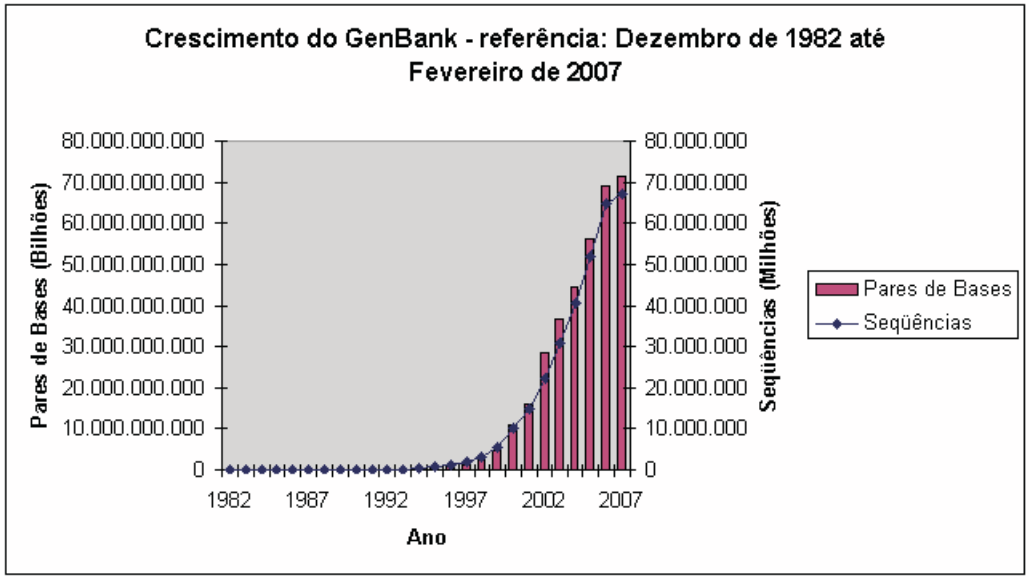


Figura 2.13: Crescimento exponencial apresentado pelo *GenBank*.

científico e taxonomia do organismo fonte, referências bibliográficas e tabela de características biológicas tais como: regiões codificantes e sua tradução em proteína, unidades de transcrição, regiões de repetição e sítios de mutação ou modificação. A maior fonte de novas seqüências são *EST*. Em relação aos registros armazenados no *GenBank*, cada um consiste de uma seqüência e sua anotação que são associados em um identificador único, o número de acesso, que permanece constante durante a "vida"do registro, mesmo quando há uma mudança em sua anotação.

A base *EMBL*, pertencente ao *European Bioinformatics Institute-EBI*, tem o propósito de coletar e apresentar seqüências de nucleotídeos e anotações, possuindo uma abrangência mundial [67]. A cobertura da base *EMBL* inclui, de modo geral, dados de projetos de seqüenciamento genômico, seqüências submetidas diretamente, registro de seqüências suportadas por aplicações patenteadas, entre outros. O principal objetivo da base é integrar as inúmeras seqüências de nucleotídeos e anotações dentro da Bioinformática. Assim como o *GenBank*, o volume de dados disponibilizados pelo *EMBL* apresenta, considerando como referência o último *release* disponibilizado, crescimento exponencial. A Figura 2.14 apresenta o gráfico de crescimento da base *EMBL*. Atualmente, a base vêm concentrado esforços no desenvolvimento de ferramentas para submissão e recuperação de dados, a fim de prover aos pesquisadores maior facilidade na submissão e maximizar a qualidade das informações contidas nos dados submetidos [5].

O banco *DDBJ* surgiu em 1986 no Instituto Nacional de Genéticas do Japão ou *National Institute of Genetics* (*NIG*), sendo que o seu primeiro *release*, contendo 66 seqüências e 54.485 pares de bases, foi divulgado em julho de 1987 [21]. O *DDBJ* é o único repositório de seqüências biológicas no Japão que é certificado oficialmente para coletar seqüências de *DNA* e emitir o número internacional de identificação aos responsáveis pelos dados submetidos. Grande parte das submissões de seqüências são oriundas de pesquisadores japoneses, chineses e

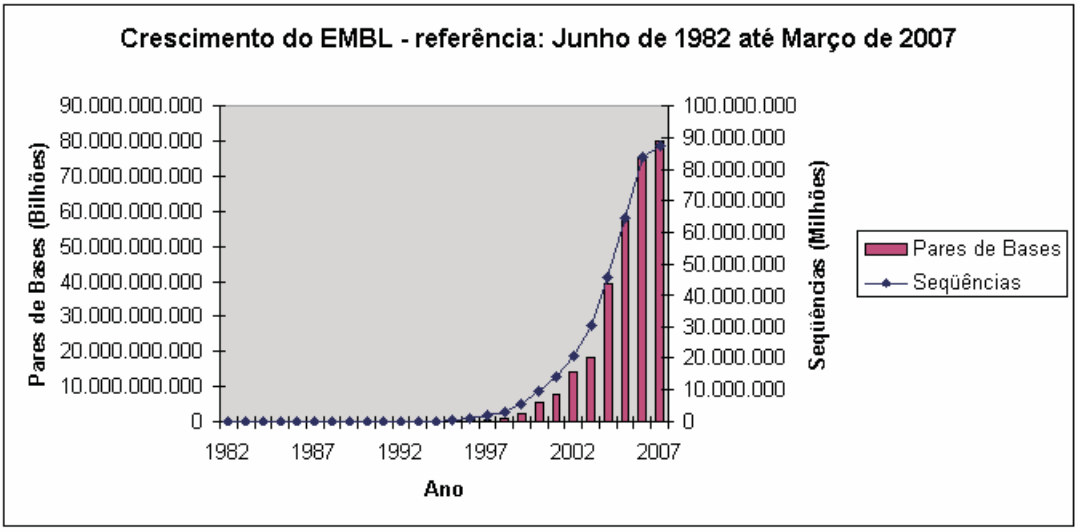


Figura 2.14: Crescimento exponencial apresentado pelo *EMBL*.

coreanos [141]. Assim como o *GenBank*, o *DDBJ* contém um grande volume de *EST*, sendo divido em categorias, tais como: genes humanos, primatas (exceto os seres humanos), roedores, vertebrados, invertebrados, bactérias, vírus, *EST*, entre outros. Os pesquisadores do *DDBJ*, da mesma forma que os grupos do *GenBank* e *EMBL*, além de outros, desenvolvem e disponibilizam *softwares* para recuperação e análise de dados biológicos. A Figura 2.15 apresenta o gráfico de crescimento da base *DDBJ*.

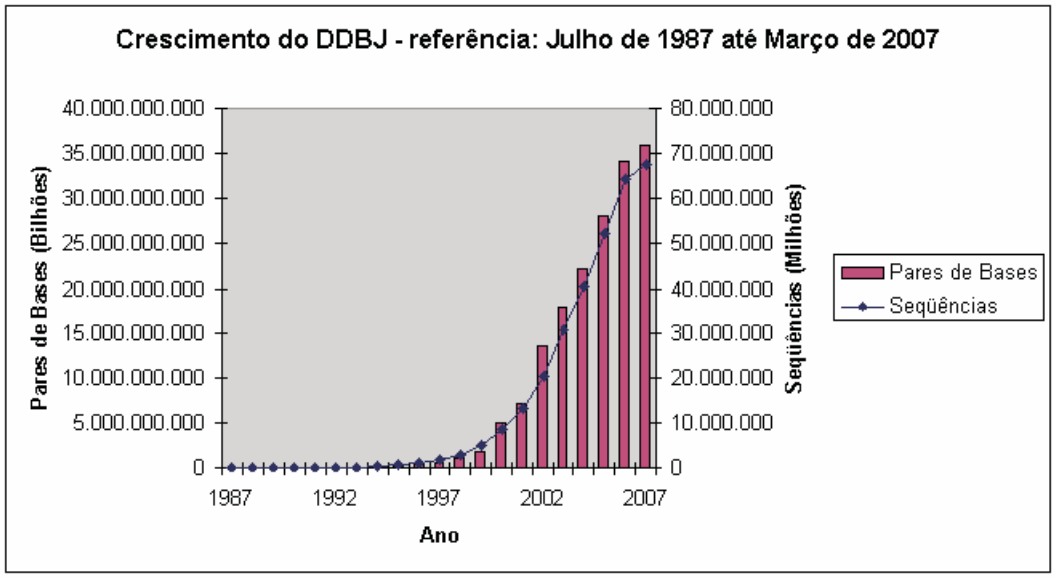


Figura 2.15: Crescimento exponencial apresentado pelo *DDBJ*.

Nr

A base *nr* é o mais amplo banco de dados de nucleotídeos disponível através do *NCBI*. A composição do *"nr"* é feita por seqüências não-redundantes, derivadas de seqüências presentes nas bases *GenBank*, *EMBL* e *DDBJ*. Deve-se observar, que além do banco com seqüências de nucleotídeos, a base *"nr"* possui um repositório para seqüências proteícas, derivadas de seqüências obtidas das bases *Swiss-Prot*, *Swiss-Prot updates*, *Protein Information Resource* (*PIR*) e *Protein Data Bank* (*PDB*) [9, 22]. Neste trabalho, a base *"nr"* utilizada refere-se à base de seqüências protéicas.

Swiss-Prot

A base de conhecimento de proteínas, *Swiss-Prot*, pertence a um repositório central conhecido como *Universal Protein Resource KnowledgeBase* (*UniProtKB*), o qual é mantido pelo consórcio *UniProt* [48]. Este consórcio é formado pelas instutuições *EBI*, *PIR* e o *Swiss Institute of Bioinformatics* (*SIB*). *Swiss-Prot* é conhecido por ser uma base de dados que possui características como: anotações de alta qualidade, dados não-redundantes, nomenclaturas padronizadas e integração com outras bases de dados, tais como *GenBank*, *EMBL* e *DDBJ*[59].

COG/KOG

A base de dados *COG* tem por objetivo classificar filogeneticamente proteínas codificadas em genomas completos de bactérias, organismos *archaea* (organismos relacionados com as bactérias) e organismos eucariotos [147]. A base de dados *COG* foi projetada para simplificar estudos evolucionários de genomas completos de procariotos, bem como de eucariotos unicelulares, e melhorar as atribuições funcionais de proteínas individuais [53]. Cada *COG* consiste de proteínas individuais ou grupos de parálogos presentes em pelo menos 3 linhagens de organismos. Com o objetivo de criar uma bases para pesquisa de genes ortólogos em diversos grupos de organismos eucariotos, foi desenvolvida a base *KOG*. A análise desta base reporta tendências presentes na evolução dos organismos eucariotos [146].

GO

O projeto *GO* objetiva prover vocabulários estruturados para domínios biológicos específicos que podem ser utilizados para descrever genes ou produtos de genes [47, 123]. O *GO* iniciou em 1998, como uma colaboração entre três bases de organismos, a saber: *FlyBase*, *Saccharomyces Genome Database* (*SGD*) e a *Mouse Genome Database* (*MGD*) [23].

GO não é uma base de dados com seqüências de genes e nem um catálogo com produtos de genes. *GO* é um projeto que descreve como produtos de genes comportam-se dentro do contexto celular, possuindo esforços colaborativos, observando a necessidade por descrições consistentes de produtos de genes em bases de dados diferentes. Em outras palavras, o Projeto *GO* visa definir e/ou uniformizar a terminologia utilizada pelos biólogos para descrever diferentes elementos de Biologia Molecular que são compartilhados entre as várias formas de vida, de

modo a viabilizar um modo padrão e universal de anotar genes ou seus produtos. Esta uniformização possibilita a análise e o cruzamento dos dados biológicos dos mais diversos organismos de forma consistente.

O projeto *GO* desenvolveu três vocabulários controlados, também conhecidos como três ontologias *GO* (a definição de ontologia é apresentada na Seção 3.4.1), que descrevem produtos de genes nos termos de seus processos biológicos associados, componentes celulares e função molecular. Em relação a esse trabalho realizado pelos pesquisadores do projeto *GO*, três aspectos devem ser considerados: (1) o desenvolvimento e a manutenção das três ontologias; (2) a anotação de produtos de genes, a qual envolve fazer associações entre as ontologias e os genes e produtos de genes em bases de dados colaborativas; (3) desenvolver ferramentas que facilitem a criação, manutenção e o uso de ontologias [23].

Pfam

Pfam é uma base de dados com uma grande coleção de alinhamentos múltiplos de seqüências e *HMM* abrangindo muitas famílias comuns de proteínas, tendo como objetivo ser uma base com definições precisas acerca de domínios protéicos [51]. Cada família de proteínas presente na base *Pfam* é representada por dois alinhamentos múltiplos de seqüências e dois perfis *HMM*.

A quantidade de famílias protéicas encontradas na base *Pfam* têm crescido bastante nos últimos anos, apresentando - no *release* 21 disponibilizado em novembro de 2006 - alinhamentos e modelos para um total de 8957 famílias, baseadas nas bases de seqüências protéicas *Swiss-Prot* [18] e *SP-TrEMBL* [24]. A base *Pfam* é subdividida em duas outras bases, conhecidas como *Pfam-A* e *Pfam-B*. As famílias protéicas presentes na base *Pfam-B* são aquelas não pertencentes à alguma família *Pfam*. A Figura 2.16 exibe as porcentagens relativas às famílias encontradas nas bases *Pfam-A* e *Pfam-B*.

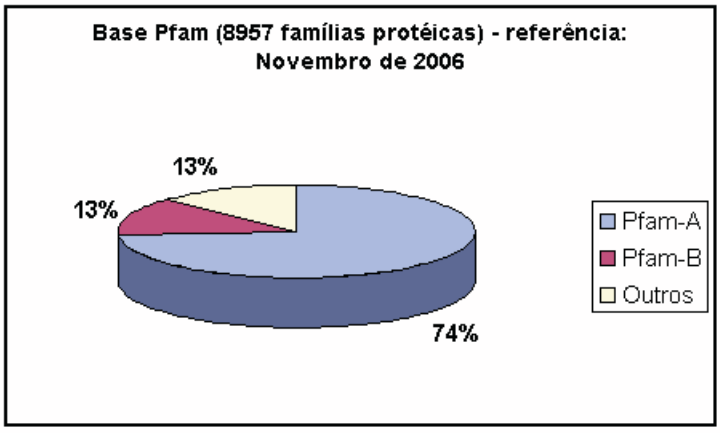


Figura 2.16: Porcentagens das bases *Pfam-A* e *Pfam-B* em relação as 8957 famílias em *Pfam* (adaptada de [6]).

Rfam

O *Rfam* é uma abrangente base de dados de alinhamentos múltiplos de seqüências e MC, cobrindo diversas famílias comuns de *RNA* não codificadores [91]. Para cada família presente na base *Rfam* que possui mais de 574 famílias - de acordo com o *release* 8.0, disponibilizado em fevereiro de 2007 - o pesquisador pode: visualizar e utilizar alinhamentos múltiplos de seqüências, ler a anotação e examinar a distribuição de espécies dos membros das famílias disponíveis, entre outras funcionalidades.

A base *Rfam*, em conjunto com o programa *INFERNAL* (descrito na Seção 2.5), pode ser utilizada para anotar seqüências, incluindo genomas completos, a partir de conhecidas seqüências de *ncRNA*. Segundo [91], um dos objetivos do *Rfam* é facilitar a identificação e classificação de novos membros, a partir de famílias de seqüências conhecidas, e disponibilizar anotações de *ncRNA*. A base *Rfam* pode ser acessada em [25].

SMART

SMART é uma base de seqüências utilizada para identificação, anotação e análise de arquiteturas de domínios protéicos [133, 103]. Os dados contidos em *SMART* são derivados de alinhamentos múltiplos de domínios protéicos. Assim como os *HMM*, tais alinhamentos permitem aos usuários a identificação de domínios em bases de seqüências. Estes domínios são freqüentemente anotados com respeito às distribuições filogenéticas, classes funcionais, estruturas terciárias (das proteínas) e resíduos funcionalmente importantes. Os domínios encontrados em bancos de seqüências não redundantes (por exemplo, a base *nr*), bem como parâmetros de consultas e informações taxonômicas, são armazenados em sistemas gerenciadores de bancos de dados relacionais. Dessa forma, é permitido aos usuários fazerem buscas a proteínas contendo combinações específicas de domínios em categorias taxonômicas definidas.

A base *SMART* pode ser utilizada de dois modos: normal e genômico. A principal diferença entre estes dois modos é a base protéica utilizada. No modo normal, são utilizados as bases *Swiss-Prot*, *SP-TrEMBL* e proteomas da base *Ensembl*. No modo genômico, apenas proteomas de genomas completamente seqüenciados são utilizados [26].

KEGG

Kyoto Encyclopedia of Genes and Genomes (*KEGG*) é uma base de conhecimento para análise sistemática de funções de genes, vinculando informações genômicas com informações funcionais [97]. A base *KEGG* é constituida de *softwares* associados e três bases de dados, ambos disponibilizados no *Japanese GenomeNet service* (Serviço Japonês de Rede Genômica) , a saber:

- A base *PATHWAY*, que contém representações gráficas de processos celulares, como: metabolismo, transporte de membrana e ciclo celular. A base é complementada por um conjunto de tabelas contendo grupos ortólogos para a informação sobre vias metabólicas de motivos, que são freqüentemente

codificados por posições de pares de genes no cromossomo e especialmente úteis na predição de funções de genes;

- A base *GENES*, abrange uma coleção de catálogos de genes para todas as seqüências genômicas completas e parciais, com anotações atuais sobre funções de genes;
- A base *LIGAND*, é uma coleção de compostos químicos celulares, bem como moléculas de enzimas e reações enzimáticas relevantes.

Capítulo 3

Sistemas Multiagente

Conforme apresentado em [58] e [145], o surgimento de arquiteturas de *software* distribuídas, juntamente com o amadurecimento de técnicas como Computação em Grade [72] e *Web Services* [144], têm contribuído fortemente para o desenvolvimento de novas tecnologias na área de Computação Distribuída. Características complementares a sistemas de computação distribuída incluem arquitetura de *software* modular para garantir a escalabilidade dos sistemas, bem como mecanismos que asseguram capacidades tais como: adaptabilidade, coordenação entre os elementos do sistema e uma estrutura comun de comunicação. Esses requisitos contribuiram no amadurecimento de metodologias e técnicas para o desenvolvimento de *softwares* orientados a agentes, provendo um conjunto de abstrações flexíveis de alto nível para modelar problemas complexos de forma distribuída. Os estudos em relação as arquiteturas de *software* baseadas no paradigma de agentes, sinalizam que problemas de natureza distribuída ou que requerem uma sinergia entre um número de elementos distribuídos para sua solução podem ser eficientemente implementados por meio de Sistemas Multiagente (SMA).

Pesquisas em SMA no domínio da Inteligência Artificial Distribuída (IAD)[1] preocupam-se com o estudo, comportamento e construção de um coleção de agentes autônomos que interagem entre si e com o seu ambiente [143]. Há vertentes relacionadas ao paradigma de agentes inteligentes na qual acredita-se que a inteligência não pode ser separada do contexto social, ou seja, não considerando os agentes apenas como entidades inteligentes isoladas. A partir deste contexto, aplica-se a idéia de SMA [85].

Nesse capítulo são apresentados conceitos, características e classificação de agentes inteligentes na Seção 3.1. A descrição de SMA, com suas características, classificações, ambientes e uma breve descrição de arquiteturas *blackboard* são apresentados na Seção 3.2. Aspectos em desenvolvimento de SMA, abrangendo uma visão geral da especificação *Foundation for Intelligent Physical Agents (FIPA)* e descrição do *framework JADE*, são apresentados na Seção 3.3. Por fim, tópicos relacionados a Ontologias e Mineração de Dados, que integram a arquitetura de SMA proposta nesta dissertação são apresentados na Seção 3.4.

[1] A IA tradicional possui a metáfora da inteligência baseada no comportamento individual humano com ênfase na representação de conhecimentos e métodos de inferência. Na IAD, a metáfora utilizada é baseada no comportamento social com ênfase nas ações e interações.

3.1 Agentes Inteligentes

Segundo [129], agente é uma entidade capaz de realizar a percepção de seu ambiente através de sensores e de agir sobre esse ambiente através de atuadores. Uma breve explanação acerca de ambientes serão descritos na Seção 3.2.1. A Figura 3.1 apresenta a arquitetura clássica de um agente, onde podemos observar a função de agente (símbolo de ?) que transforma qualquer seqüência de percepções em ações no ambiente. Para um melhor entendimento, consideramos o exemplo de um agente robótico [129]. Neste agente poderíamos ter câmeras e detectores da faixa de infravermelho funcionando como sensores e seus vários motores (por exemplo, braços robóticos e esteiras de movimentação) como atuadores.

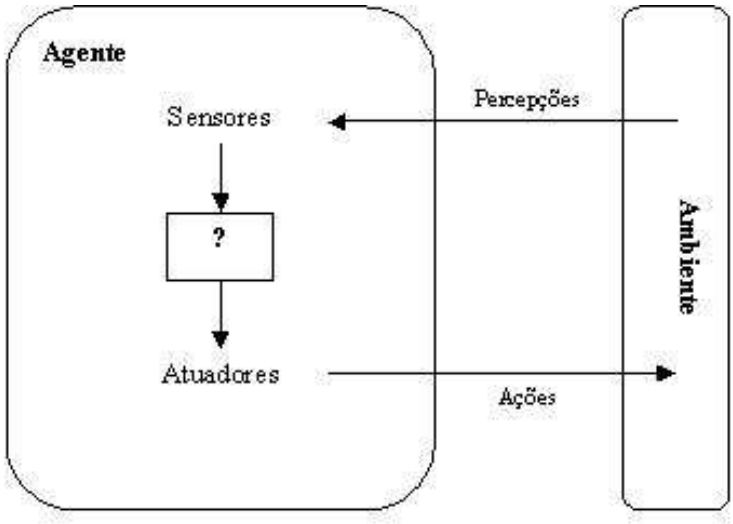


Figura 3.1: Arquitetura clássica de um agente [129].

A definição acima apenas descreve o que seja um agente, porém não engloba o significado de um agente inteligente. Vale ressaltar que vários grupos de pesquisas possuem compreensões do que venha a ser um agente inteligente, resultantes da própria definição do termo inteligência. Um agente é adequadamente classificado como inteligente quando é capaz de apresentar comportamento autônomo na execução de suas ações, ou seja, percebendo e agindo sobre o ambiente de acordo com o seu histórico de percepção adicionado a características de aprendizagem. Existem definições importantes para agentes inteligentes encontradas na literatura:

- Para [113] agentes autônomos são sistemas computacionais que habitam ambientes dinâmicos complexos, percebendo e agindo de forma autônoma neste ambiente, realizando um conjunto de objetivos e tarefas para os quais são projetados.

- [87] considera que um agente autônomo é um sistema inserido em um ambiente, o qual percebe e age sobre este ambiente, perseguindo sua própria agenda e executando suas percepções em ações futuras.

- Segundo [157] um agente inteligente é um agente capaz de ações autônomas flexíveis e, ainda, socialmente organizadas que podem, mas não necessariamente, serem dirigidas para objetivos predeterminados ou metas. Nesta definição, flexibilidade significa: reatividade, pró-atividade e habilidade social.

Em relação as características desejáveis aos agentes, estas podem variar de acordo com o domínio da aplicação, no entanto, observando a definição de [157], agentes inteligentes devem possuir quatro características básicas, a saber:

- Autonomia: os agentes inteligentes exercem controle sobre suas próprias ações.

- Reatividade: agentes inteligentes são capazes de perceber e responder em tempo adequado (observando a complexidade do domínio) às mudanças que ocorrem no ambiente, bem como satisfazer seus objetivos projetados.

- Pró-atividade: agentes inteligentes são capazes de exibir comportamento orientado a objetivos, tomando iniciativas visando satisfazer seus objetivos.

- Sociabilidade: a noção de sociedade, deve-se ao fato de que os agentes inteligentes podem e devem relacionar-se com os outros agentes (e possivelmente seres humanos) no ambiente no qual estão inseridos, buscando satisfazer seus objetivos.

No que se refere a classificação de agentes, [129] apresentam quatro tipos básicos, os quais incorporam os princípios subjacentes a quase todos os sistemas inteligentes, a saber:

- Agente reativo simples: é o tipo mais simples de agente. Estes agentes selecionam ações com base na percepção atual, ignorando o restante do histórico de percepções. Podemos considerar como exemplo um temporizador de um aparelho de ar condicionado, que muda seu comportamento de acordo com a variação de temperatura no ambiente.

- Agentes reativos baseado em modelos: são agentes que podem manter algum estado interno que dependa de seu histórico de percepções (refletindo pelo menos alguns dos aspectos não-observados do estado atual) e que também possam alterá-los/atualizá-los. De forma simplificada, esse tipo de agente controla o estado atual do ambiente usando um modelo interno. Em seguida, ele escolhe uma ação da mesma maneira que o agente reativo simples.

- Agentes baseados em objetivos: assim como agentes precisam de uma descrição de seus estados atuais, eles também precisam de alguma espécie de informação sobre objetivos que descreva situações desejáveis. Por exemplo, um carro em um cruzamento pode seguir em frente, ir para a esquerda ou direita, sendo que a decisão correta do sentido a seguir dependerá de onde está se querendo chegar.

- Agentes baseados na utilidade: são agentes que possuam funcionalidades que os tornem capazes de preferir um estado em detrimento de outro, fazendo com que este estado tenha maior utilidade para o agente no ambiente. A funcionalidade que prover utilidade mapeia um estado (ou uma seqüência de estados) em, por exemplo, um número real que descreve o grau de satisfação associado.

Estes quatro tipos básicos de agentes, podem ser divididos em duas classes: agentes reativos e agentes cognitivos. Os agentes reativos englobam os agentes reativos simples e baseados em modelos [84, 129]. Os agentes cognitivos englobam os agentes baseados em objetivos e na utilidade. Relacionando essas classes com as principais características apresentadas por agentes inteligentes, temos que os agentes reativos, por não agirem de acordo com seu histórico de percepção, diferem dos agentes cognitivos principalmente pela capacidade de aprendizagem, refletida na característica de autonomia. A autonomia é considerada por muitos pesquisadores como pré-requisito para considerar um agente como inteligente.

Paradigmas: Agentes *versus* Objetos

Segundo [157] e [112] os grupos de programadores tradicionais que usam o paradigma de Orientação a Objetos freqüentemente não conseguem observar algo de novo num sistema baseado em agentes. Ao observarmos propriedades relativas a agentes e a objetos, este fato é considerável. Em linhas gerais, objetos são entidades computacionais com estado encapsulado, podendo executar ações através de métodos associados a este estado e comunicam-se através de envio de mensagens. As diferenças entre agentes e objetos incluem o fato de que objetos raramente exibem controle sobre seu próprio comportamento (autonomia). Os agentes decidem entre si se devem ou não executar uma ação ou pedido de um outro agente. Agentes não invocam métodos entre si, mas sim requerem ações que devam ser executadas. Agentes são projetados para terem comportamento flexível, ou seja, comportamento reativo, proativo e social. Agentes interativos possuem suas próprias linhas de controle (*threads*). Contudo, devemos observar que linguagens de programação voltadas a orientação a objetos podem ser utilizadas na construção de sistemas de agentes. Como exemplo, temos a linguagem *Java* [27] que tem sido bastante utilizada na implementação de *frameworks* para construção de SMA, tais como o *JADE* [28] e o *ZEUS* [29].

3.2 Aspectos de SMA

Segundo [157] um SMA consiste de um número de agentes autônomos que interagem entre si, de acordo com os seus objetivos, tipicamente por envio de mensagens por meio de uma infra-estrutura de comunicação. De acordo com [145] são três as características centrais dos SMA:

- os agentes que estão dispersos pelo ambiente agem de forma autônoma ou em colaboração;
- os agentes que compõem o SMA não possuem o controle global do sistema;

- os agentes são providos de uma infra-estrutura para a especificação da comunicação com protocolos de troca de mensagens.

No que se refere a infra-estrutura para especificação da comunicação, temos o exemplo de arquiteturas *blackboard* (Seção 3.2.3). Em relação aos protocolos para troca de mensagens, podemos citar como exemplo as linguagens *Knowledge Query Manipulation Language (KQML)* [152, 30] e *FIPA - Agents Communication Language (FIPA-ACL)* [31].

Após apresentadas a definição e as características desejáveis aos SMA, é importante observarmos a aplicabilidade de sistemas baseados em agentes. Em [157], são abordados pontos importantes que tornam apropriadas a aplicação de soluções baseadas em agentes, a saber:

- O ambiente é aberto, dinâmico, incerto ou complexo: nestes ambientes, sistemas capazes de ações autônomas flexíveis são freqüentemente uma ótima solução.

- Agentes são uma metáfora natural: muitos ambientes (incluindo organizações e ambientes comerciais) são modelados naturalmente como sociedades de agentes, em que há a cooperação ou senão a competição entre eles para resolver problemas complexos.

- Distribuição de dados, controle ou *expertise*: em vários ambientes, a distribuição de dados, controles ou *expertise* significa que aplicar uma solução centralizada é extremamente difícil ou, na pior das hipóteses, impossível. SMA têm ganhado espaço na solução desses problemas pela flexibilidade que eles oferecem, onde cada entidade autônoma exerce o controle de operações projetadas sobre cada base de dados.

- Sistemas Legados: um problema crescente enfrentado pelos desenvolvedores envolve os sistemas legados, sistemas que podem possuir tecnologias obsoletas, mas que são essenciais às organizações que os utilizam. A esses sistemas freqüentemente são requisitados interações com outros componentes de *software*, os quais possivelmente nunca foram imaginados pelos desenvolvedores originais. Uma solução para este problema é envolver componentes legados, desenvolvendo uma "camada de agentes"funcionais para possibilitar aos sistemas legados cooperar e comunicar-se com outros componentes de *software*.

Em suma, SMA são ideais para representar domínios que incluem múltiplos métodos de solução de problemas, pontos de vista e entidades. Os SMA são capazes de oferecer vantagens da solução de problemas de forma concorrente e distribuída, juntamente com as vantagens dos esquemas sofisticados de interação. Por interações, temos a inclusão da cooperação para alcançar um objetivo comum, a coordenação na organização das atividades para a solução do problema e a negociação de restrições de subproblemas, de modo que se alcance desempenho satisfatório [112].

3.2.1 Ambientes

Um aspecto importante para projetos de SMA é a definição do ambiente em que os agentes estão inseridos. [153] apresenta a seguinte definição para este termo: Um meio que provê as condições delimitantes para os agentes existirem e que coordena a interação entre eles com acesso aos recursos. Neste contexto, o termo recursos pode ser entendido como recursos computacionais, *hardware* e *software*.

Para um melhor entendimento do conceito de ambiente, a Tabela 3.1 exibe alguns exemplos de SMA, apresentando informações sobre medida de desempenho, ambiente, sensores e atuadores.

Tabela 3.1: Exemplos de SMA e seus ambientes (adaptada de [129]).

Descrição do SMA	**Medida de Desempenho**	**Ambiente**	**Atuadores**	**Sensores**
Robô de seleção de peças	Porcentagem de peças em bandejas corretas	Correia transportadora com peças; bandejas	Braço e mãos articulados	Câmera, sensores angulares articulados
Instrutor de inglês	Maximizar nota de aluno em teste	Conjunto de aluno, testes de agência certificadora	Exibir exercícios, sugestões, correções	Entrada pelo teclado
Anotação manual em projetos de seqüenciamento de genomas	Porcentagem de sugestões corretas, acurácia das anotações manuais	Anotadores, bases de dados de seqüências, anotações manuais	Exibir sugestões de anotações manuais	Solicitação dos anotadores

Segundo [153], alguns grupos de pesquisa consideram o ambiente de SMA como uma mistura que envolve recursos, serviços, bases de dados, infra-estrutura de comunicação, entre outros. Portanto, todos os elementos não-agentes de um SMA são tipicamente considerados como sendo parte do ambiente.

A Figura 3.2 mostra uma descrição acerca deste contexto, onde são visualizadas as posições dos agentes no ambiente, bem como sua divisão em três camadas: aplicação multiagente, plataforma de execução e infra-estrutura física. A camada da aplicação SMA abrange as aplicações de agentes, ambientes da aplicação (por exemplo, a metodologia *GAIA* de modelagem de agentes [157, 158] e *framework* SMA, por exemplo, *JADE* [55, 56]). A camada da plataforma de execução abrange *middlewares*, por exemplo, *Remote Method Invocation* (*RMI*), e o sistema operacional. Por último, na camada infra-estrutura física estão os *hardwares* dos computadores, redes de comunicação e um mundo físico, por exemplo, celulares e *Personal Digital Assistants* (*PDA*).

Em relação à classificação dos ambientes de SMA, [129] divide e define como completamente ou parcialmente observável, determinístico ou estocástico, episódico ou seqüencial, estático ou dinâmico, discreto ou contínuo. Note que essas classi-

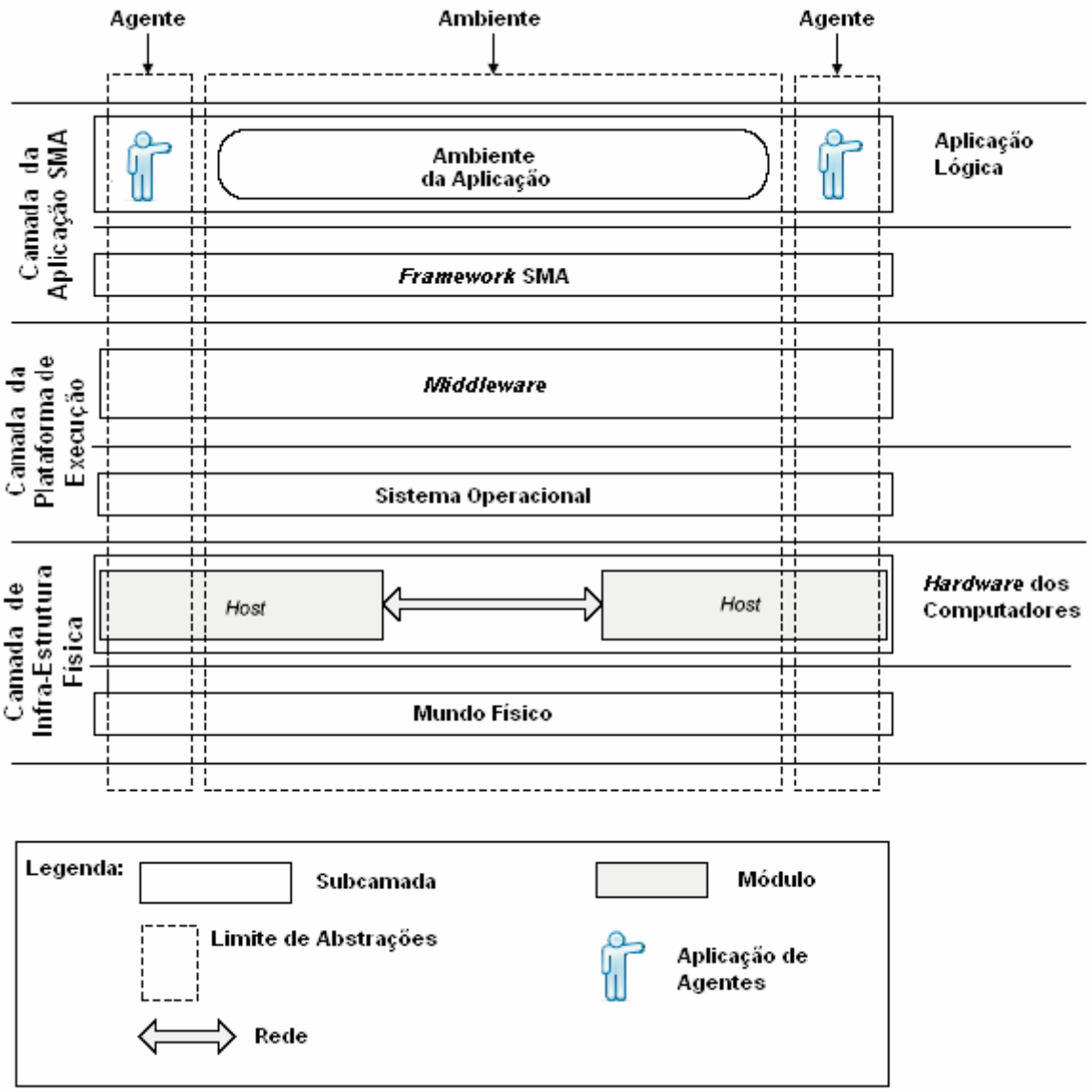


Figura 3.2: Ambiente divido em três camadas (adaptado de [154]).

ficações não são mutualmente excludentes, já que um ambiente pode possuir mais de uma dessas classificações.

- Completamente observável ou parcialmente observável: se os sensores de um agente permitem acesso ao estado completo do ambiente em cada instante, o ambiente é completamente observável. Um ambiente poderia ser parcialmente observável devido aos agentes apresentarem sensores imprecisos.

- Determinístico ou estocástico: se o próximo estado do ambiente é completamente determinado pelo estado atual e pela ação executada pelo agente, o ambiente é classificado como determinístico; caso contrário, ele é estocástico.

- Episódico ou seqüencial: em ambientes episódicos, a experiência do agente é dividida em episódios atômicos, onde cada episódio consiste na percepção do agente, e depois na execução de uma única ação. O episódio seguinte não pode depender das ações executadas em episódios anteriores. No caso de ambientes seqüencias, a decisão atual pode afetar todas as decisões futuras.

- Estático ou dinâmico: se for possível o ambiente sofrer alterações enquanto um agente está deliberando, o ambiente, na visão deste agente, é dinâmico, caso contrário, ele é estático.

- Discreto versus contínuo: o ambiente é considerado discreto se existe um número finito de estados distintos, percepções e ações possíveis, caso contrário ele é considerado contínuo.

3.2.2 Classificação

Os SMA podem ser classificados como SMA reativos e cognitivos. Os SMA reativos baseiam-se na idéia de que o comportamento inteligente deve emergir da interação de diversos agentes simples. SMA reativos consideram o problema, para o qual estão sendo utilizados, como sendo um conjunto de agentes interagindo entre si, onde cada agente possui seus próprios objetivos. Uma forma usual de representar os comportamentos dos agentes é por meio de um conjunto de regras (como, regras de produção). As principais características dos SMA reativos, conforme cita [122, 41] são:

- não há representação explícita de conhecimento: o conhecimento do agente é implícito através de regras de comportamento que se manifestam em suas ações comportamentais.

- não há representação explícita do ambiente: o comportamento (resposta) do agente baseia-se no que é percebido (estímulo) a cada instante, mas sem uma representação explícita do ambiente.

- não há memória de ações: os agentes não mantém um histórico de suas ações, de forma que uma ação passada não exerce nenhuma influência direta sobre suas ações futuras.

- possue um grande número de membros: este tipo de SMA tem, em geral, um grande número de agentes, variando de dezenas a milhões de agentes dependendo do domínio da aplicação.

Pesquisando vários autores observamos que ao contrário da relação de agentes que compõem os SMA reativos, onde os agentes exibem comportamentos do tipo estímulo-resposta, os SMA cognitivos podem ser descritos como um conjunto de agentes socialmente organizados capazes de interagir (cooperar, coordenar e negociar) com os demais agentes da sociedade. Desta forma, podemos perceber que os SMA cognitivos possuem uma modelagem mais complexa se comparado aos SMA reativos [84, 152, 46, 129, 122, 41].

Algumas características aplicadas aos SMA cognitivos são apresentadas conforme [122, 41]:

- existe representação explícita do conhecimento do ambiente;
- é mantida um histórico de percepções passadas, objetivando o planejamento de ações futuras;
- são compostos de um pequeno número de membros;
- não há um sistema de controle global;
- possuem informações e capacidades incompletas para a solução do problema;
- existe uma computação assíncrona.

Relacionando os dois tipos de classificação apresentados ao contexto deste trabalho, na proposta que será apresentada no Capítulo 4 temos a arquitetura baseada em SMA cognitivo, principalmente pela necessidade dos agentes serem capazes de melhorar as sugestões de anotações manuais a partir de um modelo de aprendizagem específico, auxiliado por ontologias próprias e algoritmos de mineração de dados. As definições relacionadas a estes tópicos são apresentadas na Seção 3.4. Entretanto, o protótipo desenvolvido durante este trabalho não aplicou estes modelos de aprendizagem, mas tão somente a elaboração de um vocabulário comum aos resultados reportados pelos *softwares BLAST* e *FASTA*.

Em relação ao protótipo implementado, utilizamos o conceito de SMA reativos, onde a arquitetura é composta de agentes que utilizam uma estrutura de comunicação, com troca de mensagens por meios de vocabulários específicos de comunicação e representação de estruturas biológicas, baseada em uma arquitetura *blackboard* (Seção 3.2.3). Em relação a *expertise* que determina a execução das ações do agentes, a forma de representação utilizada foi de regras de produção. Os aspectos relacionados a proposta e a arquitetura implementada serão apresentados com maior nível de detalhes no Capítulo 4.

3.2.3 Arquitetura *Blackboard*

No início da década de 70 foram dados passos relevantes para formalizar a idéia presente na IAD, como a construção de sistemas inteligentes distribuídos, com o

desenvolvimento de uma arquitetura conhecida como *Blackboard* ou Quadro-negro [121, 70, 71].

A arquitetura *blackboard* provê aos sistemas uma arquitetura modular que possa ser adequada a domínios que possuem características tais como solução distribuída e *expertise* descentralizada, ou seja, a *expertise* do sistema pode ser divida em módulos de conhecimento independentes. Segundo [71], as arquiteturas *blackboard* possuem três componentes principais, a saber:

- Fonte de Conhecimento (FC): módulo independente que provê a *expertise* necessária para a solução do problema. Uma FC não interage diretamente com outra FC, bem como não possue conhecimento de outras FC presentes no sistema. A FC pode ser representada por paradigmas como, por exemplo, sistemas baseados em regras e algoritmos de aprendizagem de máquina (árvores de decisão, redes neurais, lógica *fuzzy*, redes bayesianas, entre outros).

- *Blackboard (BB)*: repositório compartilhado de dados que contêm vocabulários comuns ao sistema, dados de entrada, soluções parciais, sugestões e outras informações que possam contribuir para a solução do problema. As FC interagem diretamente com o módulo BB.

- Componente de Controle (CC): responsável por gerenciar centralizadamente o fluxo de execução para a solução do problema. O CC é um módulo separado de uma FC individual, determinando em um dado momento, qual FC é a mais apropriada para ser executada.

A Figura 3.3 mostra um esquema simples de uma arquitetura *blackboard*. As arquiteturas *blackboards*, assim como arquiteturas SMA, possibilitam o desenvolvimento de sistemas colaborativos. No desenvolvimento de um SMA é possível utilizar o modelo de arquitetura *blackboard*, no entanto, essas duas metodologias possuem características diferentes. Os SMA direcionam para características como autonomia (controle local e controle global), interação entre agentes (ao contrário da não interação entre as FCs), distribuição dos dados (ao contrário de um repositório central compartilhado como o BB), conforme apresentado na Seção 3.2. No contexto deste trabalho, o protótipo *BioAgents* apresentado no Capítulo 4 foi implementado sob o paradigma multiagente apresentando em sua arquitetura uma organização com características da abordagem *blackboard*.

3.3 Desenvolvimento de SMA

De acordo com as características inerentes aos SMA apresentadas na Seção 3.2, podemos observar que o desenvolvimento destes sistemas requer um nível de complexidade maior do que no desenvolvimento de *softwares* tradicionais, o que implica no melhoramento e surgimento de tecnologias de suporte. Observando que muitas das características básicas dos SMA são independentes de aplicação e buscando facilitar o desenvolvimento destes sistemas, diversos *frameworks* para suporte ao desenvolvimento de SMA começaram a surgir.

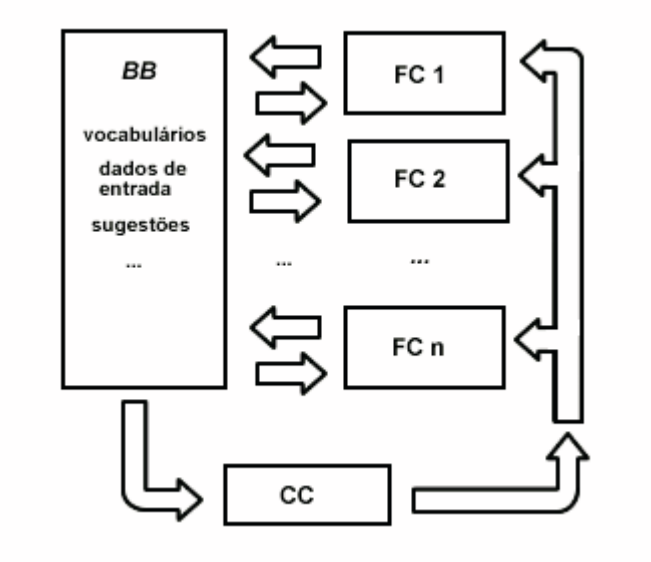


Figura 3.3: Esquema simples de uma arquitetura *blackboard* (adaptada de [70]).

Estes *frameworks* oferecem todas as funcionalidades básicas para a operabilidade de um SMA, provendo mecanismos para a gerência, comunicação, controle e coordenação, o que permite aos desenvolvedores se dedicarem por mais tempo à modelagem e funcionalidade dos agentes. A essa abordagem, dá-se o nome de abordagem horizontal ou *middleware*, ou seja, os *frameworks* oferecem uma biblioteca de nível relativamente alto, porém genérica e independente de aplicação, diferente da abordagem vertical onde soluções *ad hoc* específicas são implementadas.

3.3.1 Visão Geral sobre a Especificação *FIPA*

A *Foundation for Intelligent Physical Agents* (*FIPA*) [31] é uma organização pertencente ao *Institute of Electrical and Electronics Engineers* (*IEEE*) [32] que especifica padrões para o desenvolvimento de tecnologias baseadas em agentes inteligentes. Criada em 1996 e tendo sido aceita oficialmente em 2005 pela *IEEE* foi definida como um comitê de padrões para tecnologias de agentes, com o propósito de produzir especificações de padrões de *software* para agentes heterogêneos e sistemas baseados em agentes.

A principal missão da *FIPA* é a promoção e aprimoramento contínuo de tecnologias e especificações que facilitem e promovam a interligação de *softwares* baseados em agentes inteligentes nos setores industrial e comercial, visando à interoperabilidade entre sistemas autônomos. O trabalho de padronização da *FIPA* destina-se a facilitar a interoperabilidade entre *softwares* de agentes, uma vez que, além da linguagem de comunicação, a *FIPA* especifica também quais os principais agentes envolvidos na gestão de um sistema, a ontologia necessária para a interação entre sistemas, e ainda o nível de transporte dos protocolos.

Segundo [31, 66], no padrão *FIPA* um agente é uma entidade de *software* que encapsula seu próprio estado, comportamento, processo de controle de execução

e a habilidade de interagir e se comunicar com outras entidades. Um atributo muito interessante é a habilidade de um agente migrar de uma plataforma para a outra mantendo intacto o estado da sua informação, ou seja, o agente pode ser móvel. Entretanto, não significa que este atributo seja obrigatório à modelagem de agentes. A arquitetura da plataforma de agentes *FIPA* pode ser encontrada nas especificações normativas da Especificação FIPA de Gerenciamento de Agentes (*FIPA Agent Management Specification*) [31]. A plataforma de agentes *FIPA* provê a infra-estrutura na qual os agentes podem ser desenvolvidos, estabelecendo um modelo de referência lógico para criação, registro, localização, comunicação, migração e retirada dos agentes. A Figura 3.4 ilustra este modelo de referência, o qual é constituído dos seguintes componentes lógicos:

- Agente ou *Agent*: é o fator fundamental numa plataforma de agentes que combina um ou mais capacidades de serviços e pode incluir acesso à *softwares* externos, usuários humanos e meios de comunicação. Um agente deve ter pelo menos um proprietário e pode suportar várias noções de identidade. Por exemplo, um *Agent IDentifier* (*AID*) rotula um agente. Logo, este agente pode se distinguir dos demais sem risco de ambigüidade no seu universo.

- Facilitador de Diretórios ou *Directory Facilitator* (*DF*): é um componente obrigatório da plataforma de agentes. Ele provê um serviço de páginas amarelas (lista os serviços oferecidos) para os outros agentes. Os agentes podem registrar seus serviços com o *DF* ou requisitar ao *DF* que ele encontre serviços oferecidos pelos outros agentes. Múltiplos *DFs* podem existir dentro de uma mesma plataforma de agentes e podem estar confederados.

- Sistema de Gerenciamento de Agente ou *Agent Management System* (*AMS*): é um componente obrigatório da plataforma de agentes. O *AMS* exerce um controle superintendente de acesso e uso da plataforma. Só pode existir um *AMS* em uma única plataforma de agentes. O *AMS* apresenta um diretório de *AIDs* que contém endereços de transporte para agentes registrados na plataforma. O *AMS* oferece páginas brancas para os outros agentes, isto é, permite a um agente encontrar agentes capazes de prover um determinado serviço de que necessita. Cada agente deve registrar-se com o *AMS* de forma a receber uma *AID* válida.

- Sistema de Transporte de Mensagens ou *Message Transport System* (*MTS*): é o método de comunicação padrão entre agentes de diferentes plataformas.

- Plataforma de Agentes ou *Agent Platform* (*AP*): provê a estrutura física em que o agente pode ser desenvolvido. A *AP* é composta de máquina(s), sistema operacional, *software* de suporte de agentes (por exemplo, *JADE*), componentes de gerenciamento de agentes *FIPA* (composto de *DF*, *AMS* e *MTS*) e agentes. O mecanismo interno da *AP* é inerente ao desenvolvedor do SMA e não há uma padronização dentro do padrão *FIPA*. O *FIPA* está interessado somente em como a comunicação é feita entre agentes que são

nativos da *AP* e fora dela ou àqueles que podem dinamicamente se registrar com uma *AP*. Agentes são livres para trocar mensagens diretamente utilizando quaisquer meios que eles possam suportar.

- *Software*: descreve todos os não-agentes, coleção de instruções executáveis acessíveis através de um agente. O agente pode acessar o *software* para adicionar novos serviços, adquirir novos protocolos de comunicação, entre outros.

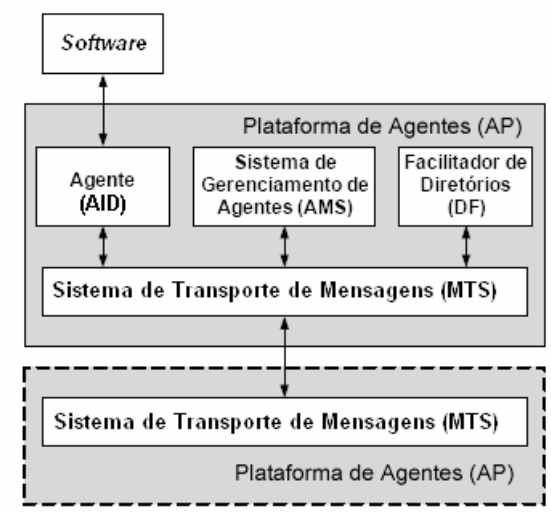


Figura 3.4: Modelo de referência *FIPA* para a plataforma de agentes (adaptada de [66]).

Em relação ao atributo comunicação, seu gerenciamento é um fator determinante para a qualidade de sistemas com o objetivo de prover suporte ao desenvolvimento de SMA. Um dos padrões desenvolvidos pela *FIPA* é a linguagem de comunicação de agentes *FIPA-ACL* [31], conforme é comentado na Seção 3.2. Este padrão define o mínimo de conjuntos de tipos de mensagens, chamados de atos comunicativos.

Cada ato comunicativo é descrito entre uma forma narrativa e uma forma semântica baseada em um modelo lógico, representando a vontade de um agente sobre determinada informação carregada pela mensagem. Em [89], alguns exemplos de atos comunicativos são: *query-if* (ação de perguntar a um agente se uma proposição é verdadeira ou falsa), *inform* (o agente emissor da mensagem informa ao agente receptor se uma dada proposição é verdadeira), *not-understood* (o agente emissor informa ao agente receptor que não entendeu uma ação ou ato prévio do receptor), *agree* (informa a concordância em executar alguma ação, possivelmente no futuro), *request* (o agente emissor solicita ao receptor que ele execute alguma ação). A especificação de todos os atos comunicativos é encontrada em [31].

As mensagens da linguagem *FIPA - ACL*, em sua grande maioria, são compostas de atributos como o ato comunicativo (conhecido como *performative*), um

agente emissor, um agente receptor, um campo de conteúdo e o controle de conversação [31, 89]. O código abaixo retirado de [89] mostra exemplos de troca de mensagens entre dois agentes utilizando a linguagem de comunicação *FIPA - ACL*. Observe que no exemplo (a), o ato comunicativo é indicado por `query-if`, o emissor, receptor e conteúdo são indicados, respectivamente, pelos campos `sender`, `receiver` e `content`, por último, o controle de conversação é indicado pelo rótulo `reply-with` (identificador da mensagem). No exemplo (b) a estrutura é análoga. Em (a) o agente i pergunta ao agente j se este está registrado com o domínio d1, e em (b) o agente j responde não.

```
(a) (query-if
    :sender (agent-identifier :name i)
    :receiver (set (agent-identifier :name j))
    :content  "((registered (server d1) (agent j)))"
    :reply-with r09
    )

(b) (inform
    :sender (agent-identifier :name j)
    :receiver (set (agent-identifier :name i))
    :content  "((not (registered (server d1) (agent j))))"
    :in-reply-to r09
    )
```

No que se refere ao desenvolvimento de SMA, no *framework JADE* (Seção 3.3.2) os programadores podem acessar os atributos das mensagens por métodos pertencentes a classe que implementa a especificação da estrutura das mensagens *FIPA-ACL*, no caso, a classe *ACLMessage* [28].

3.3.2 O *framework JADE*

Segundo [55] o *JADE*, desenvolvido e distribuído pela *TILAB* (*Telecom Italia LAB*) sob a licença *Lesser GNU Public License (LGPL)*[2], é um *framework* para desenvolvimento e execução de aplicações *peer-to-peer* baseadas em agentes (reativos e/ou cognitivos), que pode operar tanto em ambientes computacionais *wired* como *wireless*. O objetivo principal do *JADE* é facilitar o desenvolvimento de *softwares* baseados em agentes.

Desde maio de 2003, um comitê foi criado com o objetivo de supervisionar o gerenciamento do Projeto *JADE*. Atualmente esse comitê possui cinco membros, a saber: *TILAB*, *Motorola*, *Whitestein Technologies AG.*, *Profactor GmbH* e a *France Telecom RD*. A última versão oficial do *framework* é a 3.4.1, lançada em novembro de 2006. Para o desenvolvimento do protótipo descrito nesta dissertação utilizamos a versão 3.4.1.

[2]A *LGPL* é uma extensão da licença *GPL*. A licença *LGPL* é destinada especialmente a *softwares* baseados em bibliotecas de funções - por exemplo, *Application Programming Interfaces (APIs)* - para desenvolvimento de aplicações, permitindo aos desenvolvedores utilizar tais bibliotecas (códigos livres) no desenvolvimento de *softwares* proprietários (códigos não-livres) [33].

De acordo com o domínio de aplicação, um ambiente computacional pode evoluir dinamicamente com sistemas. Para o *JADE*, tais sistemas são agentes migrados, ou não, entre plataformas (considerando que sigam os padrões *FIPA* de interoperabilidade) de acordo com as necessidades da aplicação que está sendo executada no ambiente computacional.

Conforme cita [55] *JADE* é um *framework* completamente desenvolvido na linguagem *Java* e está fundamentado de acordo com os seguintes princípios:

- Interoperabilidade: conformidade com a especificação *FIPA*, o que significa que os agentes construídos sob o *JADE* podem interoperar com agentes desenvolvidos sob outros *frameworks* de desenvolvimento desde que obdeçam a especificação *FIPA*.

- Uniformidade e Portabilidade: *JADE* provê um conjunto homogêneo de *APIs* que é independente da rede e da versão *Java* utilizada. Em relação a linguagem *Java*, as *APIs JADE* fornecem recursos para desenvolvedores em ambientes *Java 2 Platform Standard Edition (J2SE)*, *Java 2 Platform Entreprise Edition (J2EE)* e *Java 2 Platform Micro Edition (J2ME)* [3].

- Facilidade de uso: apesar do *framework* possuir uma determinada complexidade, esta característica fica transparente ao usuário devido a simplicidade de uso disponibilizada pelo seu conjunto de *APIs* e a ampla documentação.

- Filosofia *Pay-as-you-go*: os desenvolvedores não precisam fazer uso de todas os recursos fornecidos pelo *framework*. Os recursos que não estão sendo utilizados, não trazem *overhead* ao desempenho da máquina.

Principais características e arquitetura do *JADE*

Segundo [55] a construção do *framework JADE* contemplou tanto o desenvolvimento de pacotes *Java* com funcionalidades prontas pra uso, quanto o desenvolvimento de *interfaces* abstratas para serem adaptadas conforme as funcionalidades requeridas para determinada aplicação de agentes. Ou seja, *JADE* oferece uma *API* para desenvolvimento de agentes em *Java* e uma plataforma distribuída de agentes em conformidade com o padrão *FIPA*. Toda comunicação entre agentes sob o *framework JADE* é feita através de trocas de mensagens.

Uma outra característica do *framework* é que sua plataforma de agentes pode ser distribuída por vários *hosts* - por exemplo, cada computador conectado a uma *Local Area Network (LAN)* ou dispositivos *PDA* conectados a uma rede *wireless* - cada um deles executando apenas uma *Java Virtual Machine* (*JVM*). Os agentes são implementados como *threads Java* e inseridos dentro de repositórios de agentes conhecidos como *containers*, que disponibilizam todo o suporte para a execução do agente e representam o ambiente de execução das aplicações de agentes. A Figura 3.5 mostra a arquitetura do *framework JADE*.

[3] *J2SE* é um ambiente de desenvolvimento para aplicações, por exemplo, em *desktop* (*standalone*). *J2EE* é um ambiente de desenvolvimento para, por exemplo, aplicações corporativas multi-camadas e aplicações *Web*. Por fim, *J2ME* é um ambiente de desenvolvimento para aplicações, por exemplo, em redes *wireless* e dispositivos móveis

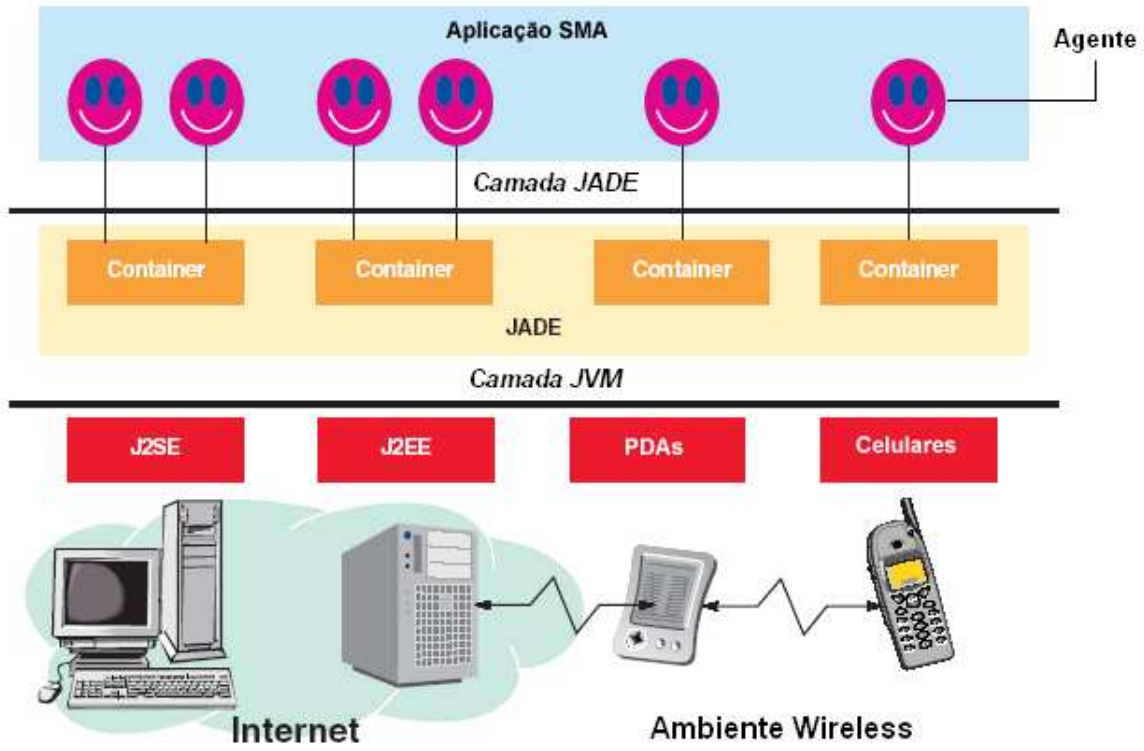


Figura 3.5: Arquitetura do *framework JADE* (adptada de [55]).

O *framework JADE* possui uma vasta biblioteca, através de um conjunto de *APIs Java*, para o desenvolvimento de agentes, bem como um ambiente de execução que prover serviços básicos, os quais devem ser ativados no dispositivo antes dos agentes serem executados (pode ser um dispositivo móvel com *J2ME* ou uma aplicação executando em um *desktop* com *J2SE* ou *J2EE*). No *JADE*, cada instância em tempo de execução é chamada de *container* (desde que contenha agentes). Ao conjunto de *containers* dar-se o nome de plataforma e esta provê uma camada homogênea na qual fica transparente aos agentes e aos desenvolvedores a complexidade existente entre o *framework* e recursos tais como sistema operacional, *hardware*, *middleware* (por exemplo, a *JVM*) e o tipo de rede de interligação. O *container* equivale a um processo. Podemos ter diferentes *containers* na mesma plataforma, sendo uma *JVM* por *container*. Além disso diferentes agentes podem existir no mesmo *container*, onde cada agente tem sua própria *thread* de execução e a *JVM* escalona um ambiente *multithreaded* preemptivo (programa modelado com trechos de código - *threads* - que executam de maneira simultânea [27]). A comunicação entre *JVMs* é feita através do método *Java Remote Method Invocation* (*RMI*) [27], recurso oferecido pela linguagem *Java* [28, 55, 56].

Na Figura 3.6 é mostrada a plataforma *JADE* distribuída em vários *containers*, sendo que há um *container* em cada um dos três *hosts* e cada *container* contêm três agentes. No *container* principal, é feito o gerenciamento de serviços tais como *AMS*, *DF* e o registro de *containers* (método *RMI*).

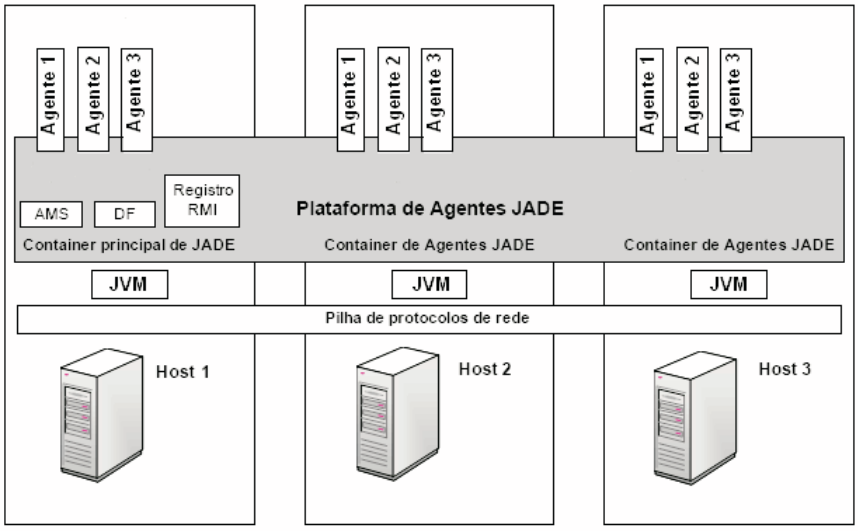


Figura 3.6: Plataforma *JADE* distribuída em vários *containers* (adaptada de [56]).

No contexto do trabalho descrito nesta dissertação, a utilização do *framework JADE* no desenvolvimento do protótipo deve-se a diversos fatores, a saber:

- ser distribuído como *software* livre, sob licença *LGPL*;
- a linguagem de programação suportada ser *Java*, possibilitando boa portabilidade;
- as especificações de *JADE* serem compatíveis com o padrão *FIPA*, oferecendo uma biblioteca de classes de protocolos de interação padronizados e prontas para serem instanciadas ou estendidas;
- não apresentar necessidade de implementar a plataforma de agentes, as funcionalidades e a ontologia de gerenciamento de agentes, nem os mecanismos de transporte e *parsing* de mensagens;
- oferecer um transporte eficiente de mensagens entre os agentes pelo uso da linguagem *FIPA-ACL*;
- possuir suporte a usuários, tendo uma grande comunidade ativa de desenvolvedores e uma vasta documentação disponível para consulta.

Agentes em *JADE*

Os aspectos dos agentes em *JADE* serão apresentados conforme [55] e [56]. Os agentes sob o gerenciamento do *framework JADE* são processos que podem representar entidades autônomas independentes, possuindo uma identidade (identificador único - *AID*) e a capacidade de se comunicar com outros agentes para

atingir os seus objetivos de execução. Os agentes podem realizar múltiplas tarefas e a comunicação por troca de mensagems dar-se de maneira assíncrona.

No ponto de vista da implementação, um agente em *JADE* é uma instância da classe *Agent*, contida no pacote *jade.core* (biblioteca de classes *JADE*), ou de alguma subclasse de *Agent*. Esta classe facilita o desenvolvimento de agentes porque provê todas as características necessárias para realizar as interações básicas com a plataforma, tais como métodos para registro, configuração ou gerenciamento remoto do agente. Além disso, a estrutura da classe *Agent* também fornece o conjunto básico de métodos que podem ser chamados para implementar o comportamento (*behavior*) personalizado do agente, por exemplo, o envio de mensagens e a utilização de protocolos de interação.

Em relação a serviços ou funcionalidades de um agente, cada um deve ser implementado como um ou mais comportamentos. Os comportamentos de *JADE* modelam arquiteturas internas reativas, contudo, a abstração de comportamento do modelo do agente em *JADE* permite a integração de *softwares* externos para enriquecer a arquitetura do agente. Além disso, *JADE* oferece também bibliotecas para implementação de características, por exemplo, mobilidade. Estas características do *framework JADE* permitem o desenvolvimento de sistemas mais sofisticados, uma vez que é possível, além de tornar os agentes capazes de migrar entre plataformas, agregar em uma mesma aplicação, por exemplo, agentes reativos simples, agentes de aprendizado, agentes baseados em utilidade e agentes móveis.

3.4 Tópicos Relacionados ao SMA Proposto

Nessa seção, apresentaremos conceitos dos temas Ontologia e Mineração de Dados inclusos na arquitetura proposta no Capítulo 4. No entanto, como o trabalho não implementa por completo a arquitetura proposta, maiores detalhes destas tecnologias não serão apresentados.

3.4.1 Ontologia

Na literatura são encontrados diversas definições para o termo ontologia e essas definições são apresentadas de acordo com o enfoque dado ao domínio da aplicação. No contexto da IA, observando a definição encontrada em [92], ontologia é uma especificação explícita e formal de uma conceitualização compartilhada. Neste contexto:

- especificação explícita diz respeito a conceitos, propriedades, relações, funções, restrições e axiomas, sendo tais propriedades explicitamente definidas;
- o termo formal refere-se ao fato da ontologia ser interpretável por máquina;
- por conceitualização entende-se a organização do conhecimento sobre o mundo - se refere ao domínio no qual a ontologia será aplicada - em forma de entidades;

- o termo compartilhada reflete a noção de que uma ontologia captura o conhecimento apresentado não somente por um único indíviduo, mais por um grupo de indivíduos (humanos e/ou agentes).

Para [152], ontologia é como uma especificação de objetos, conceitos e relacionamentos em uma área de interesse, onde as ontologias não devem ser apenas uma taxonomia de classes (ou tipos), tendo também que descrever o relacionamento entre os termos que representam conceitos.

De acordo com [128], em SMA, para o sucesso da comunicação é necessário que os agentes possuam um vocabulário comum bem definido. No entanto, é usual que diferentes agentes possuam terminologias diferentes para o mesmo significado ou terminologias iguais para significados diferentes. Uma possível solução para este problema é o compartilhamento de uma ontologia comum entre os agentes.

Conforme citado por [101] e observando o domínio da Bioinformática, no qual este trabalho está inserido, o uso de ontologias pode ser direcionado, por exemplo, a eliminar incertezas e falhas de interpretação sobre os significados dos bancos de dados, programas e seus relacionamentos. O uso de ontologias facilita a criação de sistemas aplicados à Bioinformática e podemos destacar dois tipos de ontologias dentro desta área:

- Ontologia de Biologia Molecular: identifica e associa os conceitos da Biologia Molecular, como por exemplo, os projetos *GO* e *Tambis Ontology - TaO* [49, 50].

- Ontologia de processos de Bioinformática: define conceitos como as entradas, saídas e utilização de cada programa na análise de dados da Biologia Molecular (por exemplo, a ontologia *the myGrid ontology* definida no projeto *myGrid* inglês [34]).

Neste trabalho, observando o fato exposto por [128], onde a comunicação entre os agentes necessitam de um vocabulário comum e estruturado, bem como considerando a ontologia de processos de bioinformática como descrita em [101], a prosposta apresentada no Capítulo 4 considera o uso de ontologias, contidas em um modelo de aprendizagem específico, para modelar e relacionar os conceitos presentes na utilização dos programas executados na tarefa de anotação automática e nos dados dos resultados destes programas. No protótipo implementado, foram estruturados vocabulários comuns (presentes em um componente *blackboard*) aos agentes, no entanto, como não foram descritos os relacionamentos entre os termos destes vocabulários, neste trabalho tais vocabulários não serão reportados como ontologias propriamente ditas.

3.4.2 Mineração de Dados

Mineração de Dados é o principal passo do processo de Descoberta de Conhecimento em Base de Dados ou *Knownledge Discovery in DataBases* (*KDD*), consistindo da aplicação de análise de dados e algoritmos de mineração que produzem um conjunto particular de padrões (ou modelos) sobre os dados avaliados, com o objetivo de prover informações úteis aos especialistas [81].

O tema mineração de dados teve suas discussões iniciadas na década de 90. Atualmente, existem diversos grupos de pesquisas direcionados ao estudo deste tema. Por ser um tema abordado em diversas áreas de ensino, observa-se que o material existente sobre mineração de dados possui diferentes abordagens, dependendo da origem ou do público alvo ao qual se destina. O tema é estudado e abordado por profissionais de áreas como Estatística, Informática, Matemática, Saúde, Administração, entre outras. Cada área direciona o enfoque a abordagens específicas adequadas às suas necessidades.

Em relação aos algoritmos utilizados em mineração de dados, de acordo com [156], a base técnica destes algoritmos são esquemas desenvolvidos por uma área de pesquisa relacionada a mineração de dados, conhecida como Aprendizagem de Máquina ou *Machine Learning*. Os esquemas presentes nos algoritmos de aprendizagem de máquina, com enfoque em mineração de dados, comumente trabalham com dois tipos de aprendizagem: supervisionada e não-supervisionada.

No aprendizado supervisionado, um conjunto de dados com classes conhecidas (conjunto de treinamento) é necessário para estimar os parâmetros do modelo de classificação. Após estes parâmetros serem ajustados, o modelo pode ser utilizado para classificar automaticamente todas as novas amostras dos dados [156, 95]. Resumidamente, de acordo com [112], a aprendizagem supervisionada assume a existência de um professor, alguma medida de adequação ou outro método externo de classificação de exemplos contidos em um conjunto de treinamento. Uma classe de algoritmos bastante utilizada no aprendizado supervisionado são os algoritmos de classificação, tais como redes neurais, árvores de decisão, redes bayesianas, *support vector machine* (*svm*), entre outros.

Ao contrário, no aprendizado não-supervisionado a existência do professor é eliminada, cabendo ao algoritmo de aprendizagem a avaliação dos conceitos, ou seja, os algoritmos de aprendizagem não-supervisionada envolvem a aprendizagem de padrões na entrada, quando não são fornecidos valores de saída específicos [129, 112, 156]. Por exemplo, fazendo referência ao foco deste trabalho, um agente inteligente poderia ser capaz de sugerir anotações manuais com a função biológica de um determinado gene, sem jamais ter tido o conhecimento de funções de genes homólogos à aquele gene. São exemplos de algoritmos de aprendizagem não-supervisionada: algumas variantes de redes neurais; algoritmos de agrupamento ou *clustering* - resumidamente, encontram dados com atributos similares e os coloca em um mesmo conjunto de dados; lógica nebulosa ou *fuzzy* - aplicadas em domínios de imprecisão e incerteza; entre outros.

Na literatura, são encontrados diversos trabalhos que utilizam algoritmos de aprendizado de máquina aplicados a problemas da Bioinformática como exemplo, predição de estruturas de proteínas e localização de genes na tarefa de anotação automática, pois como trata-se de um domínio que apresenta um extenso volume de dados (como observado na Seção 2.6), o uso da técnica de mineração pode ser bastante valiosa. Em [94], são relacionados diversos problemas da Bioinformática que são tratados pela mineração de dados, apresentando técnicas e algoritmos utilizados, bem como *softwares* desenvolvidos pelos diversos grupos de pesquisa existentes. Por outro lado, ainda no domínio da Bioinformática, trabalhos que utilizam a abordagem de SMA em conjunto com algoritmos de mineração, tais como aprendizagem de máquina, não são comuns. Como exemplo, podemos citar

o projeto *ATUCG - An Agent-based environmenT for a aUtomatiC annotation of Genomes* [54, 130], onde os agentes utilizam algoritmos de classificação (aprendizado supervisionado), tais como *C4.5*, *CN2*, *T2* e *Repeated Incremental Pruning to Produce Error Reduction* (*RIPPER*) para apoiar o processo de anotação. Uma descrição deste projeto, bem como de outros trabalhos correlatos ao apresentado nesta dissertação, é mostrada na Seção 5.4.

Nesse capítulo apresentamos temas relacionados ao desenvolvimento de SMA. Observaremos que no desenvolvimento do protótipo apresentado no Capítulo 4, os conceitos explanados nas Subseções 3.2.2 e 3.2.3 (classificações de SMA e arquitetura *blackboard*) são relevantes para o entendimento do modelo de agentes e de arquitetura implementada. Não menos relevantes são as tecnologias descritas na Seção 3.3, pois o *BioAgents*, em seu projeto de implementação, fez uso do *framework JADE*. Neste contexto, as especificações para desenvolvimento de sistema baseados em agentes adotadas pela *FIPA* são essencias para o desenvolvimento de SMA utilizando aquele *framework*. Para a apresentação da arquitetura proposta no Capítulo 4, descrevemos conceitos relacionados a temas como Ontologia (Subseção 3.4.1) e Mineração de Dados (Subseção 3.4.2).

Capítulo 4

BioAgents

Conforme [75], a anotação manual garante acurácia e completude ao entendimento dos dados biológicos. Com base nesta afirmação é que consideramos a necessidade de auxiliar aos biólogos, oferencendo-lhes um recurso computacional para apoiar a tarefa da anotação manual em projetos de seqüenciamento de genomas. Para trabalhar este problema, utilizamos a abordagem Multiagente principalmente pelo fato do ambiente da aplicação apresentar características específicas adequadas ao uso desta abordagem, a saber:

- utiliza bancos de dados heterogêneos e descentralizados;
- constitui um ambiente dinâmico (por exemplo, novos tipos de dados e bases de dados com constantes alterações) e
- a tarefa de anotação manual pode ser realizada de forma independente por vários biólogos.

Nesse capítulo é apresentada a arquitetura do SMA proposto na Seção 4.1. A descrição do protótipo *BioAgents* implementado e a explanação da arquitetura com suas respectivas camadas são apresentados na Seção 4.2.

4.1 Arquitetura proposta

A arquitetura proposta neste trabalho é descrita sob uma abordagem multiagente na qual diferentes agentes cooperam entre si e interagem com o ambiente, por meio de um módulo de colaboração composto de ontologias específicas e algoritmos de mineração de dados que visam resolver possíveis conflitos a fim de sugerir o melhor conjunto de anotações aos biólogos. O objetivo da utilização de mineração de dados nesta proposta é definir padrões encontrados na tarefa de anotação automática que auxiliem na anotação manual. A Figura 4.1 apresenta esta arquitetura que consiste de um modelo composto por três camadas, a saber: Camada de *Interface*, Camada Colaborativa e a Camada Física [105].

A Camada de *Interface* permite uma interação com o ambiente da aplicação, viabilizando consultas do usuário e a visualização das sugestões de anotação feitas pelo sistema. Na Camada Colaborativa, temos o núcleo do sistema. Nesta camada, as consultas do usuário são transformadas em plano de execução para

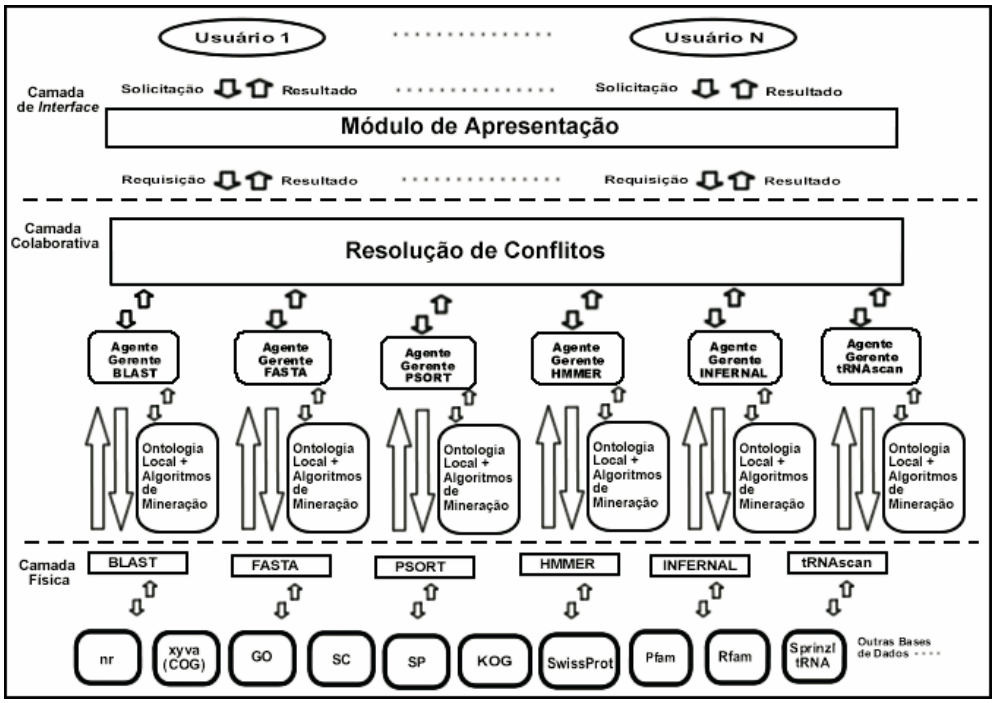


Figura 4.1: Arquitetura proposta composta por três camadas.

os agentes, definindo e avaliando quais informações extrair nas bases de dados da Camada Física, de acordo com ontologias específicas e algoritmos de mineração, combinando os resultados e resolvendo os possíveis conflitos existentes. A Camada Física é composta dos resultados da análise computacional executada na tarefa de anotação automática, isto é, as bases de dados utilizadas pela arquitetura são compostas dos arquivos de saída resultantes da análise feita pelos *softwares* descritos no interior das figuras retangulares (por exemplo, *BLAST* e *INFERNAL*) sobre as bases de dados descritas no interior das figuras elípticas (por exemplo, *nr* e *Rfam*), ou seja, as setas não indicam uma ordem lógica de execução de um determinado *software* sobre determinadas bases de dados.

No contexto desta proposta, detalhes relacionados a resolução de conflitos, ontologias específicas e algoritmos de mineração de dados não são apresentados em detalhes. Isto deve-se ao fato do protótipo apresentado nesta dissertação não consistir da implementação destas características, que estão descritas como trabalhos futuros no Capítulo 6.

4.2 O protótipo implementado

Para validar o potencial da proposta apresentada, foi desenvolvido o protótipo denominado *BioAgents*[1][107, 106]. O protótipo tem como objetivo principal auxiliar os biólogos na tarefa de anotação manual em projetos de seqüenciamento

[1]Alusão ao termo agentes biológicos (*biological agents*) comumente utilizado pelos pesquisadores na área biomédica. Exemplos de agentes biológicos são microrganismos como os vírus, bactérias e fungos.

de genomas. É importante enfatizar que no escopo deste protótipo, estamos considerando as definições de SMA reativos, conforme a classificação descrita por [84] e apresentada na Seção 3.2.2. Desta maneira, os agentes que compõem a arquitetura do protótipo *BioAgents* apresentam as seguintes características:

- não possuem representação explícita de conhecimento, ou seja, a *expertise* dos agentes é representada por módulos independentes baseados em regras de produção que direcionam o comportamento dos agentes. Sendo assim, o conhecimento provido pelos agentes é baseado no par estímulo-resposta, onde os estímulos são constituídos por percepções, refletidas pela comunicação (troca de mensagens) entre os agentes, e as respostas referem-se ao seu comportamento.

- não possuem uma representação explícita do ambiente. Nesta primeira implementação do *BioAgents*, a definição de ambiente considerada é a proposta por [153], onde o ambiente é definido como um meio que provê condições aos agentes existirem, coordenando as ações entre eles e o acesso aos recursos computacionais disponíveis.

- o comportamento dos agentes está baseado no par estímulo-resposta, indicando que suas ações atuais não exercem nenhuma influência nas ações futuras, isto é, os agentes não possuem memória de suas ações, bem como não guardam o histórico de percepções a fim de auxiliá-los na suas próximas decisões.

4.2.1 Arquitetura

O protótipo *BioAgents* é descrito sob uma arquitetura multiagente, conforme uma abordagem *blackboard* (Seção 3.2.3), composta de três camadas denominadas igualmente às camadas descritas na Seção 4.1. A Figura 4.2 apresenta a arquitetura do protótipo *BioAgents*. A organização *blackboard* implementada, conforme a figura, justifica-se devido a características como:

- a *expertise* dos agentes ser incluída em Fontes de Conhecimento (FC) específicas e independentes umas das outras. Este fato fornece um nível maior de modularidade e adequação do sistema ao tipo de projeto de seqüenciamento que o *BioAgents* será aplicado, pois ao especializar o módulo de raciocínio (*expertise*) de um determinado grupo de agentes, a acurácia do sistema pode ser melhorada sem ocasionar impacto na *expertise* dos demais agentes. Isto é explícito no domínio da anotação manual, pois esta tarefa possui a característica de *expertise* descentralizada, ou seja, a anotação manual não necessariamente é executada por um único especialista, podendo ser executada por diversos anotadores que podem apresentar níveis de experiência diferentes;

- a comunicação entre os agentes ser especificada por um repositório comum - componente *blackboard* (BB) - que contém vocabulários utilizados na comunicação e na representação dos conceitos biológicos encontrados nas bases

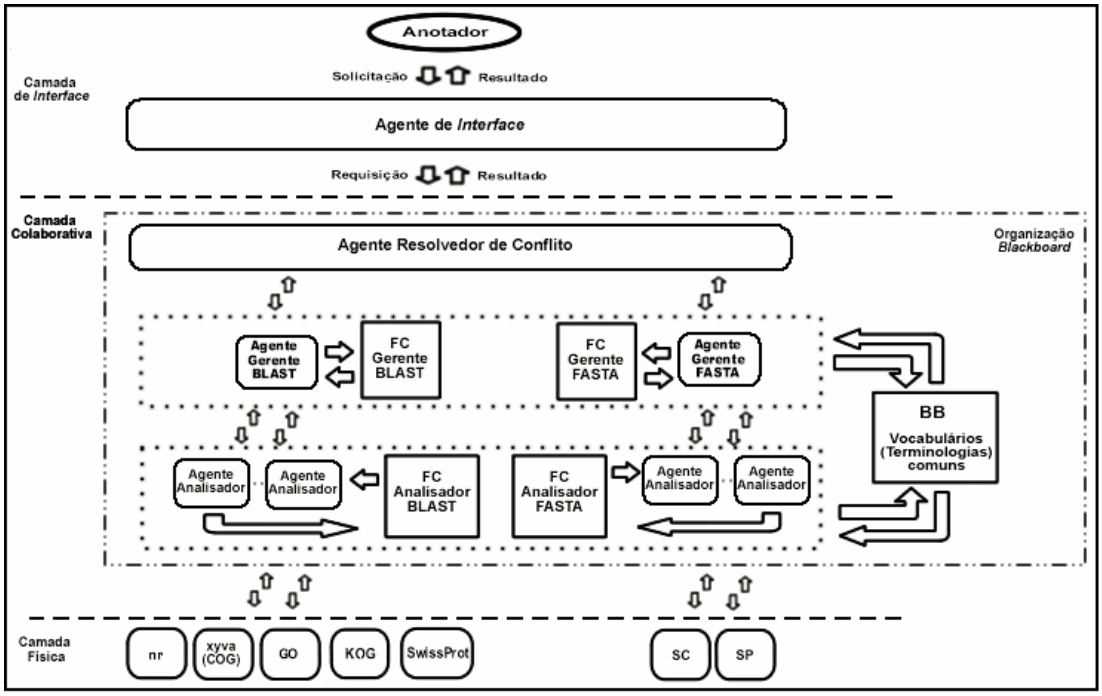


Figura 4.2: Arquitetura do *BioAgents* composta por 3 camadas.

de dados utilizadas pelo *BioAgents*. Esta estrutura é visualizada no interior do retângulo pontilhado mais externo, conforme Figura 4.2.

- o fluxo de execução dos agentes analisadores do *BioAgents* ser controlado localmente por cada agente gerente.

.

Em relação ao ambiente de desenvolvimento, o *BioAgents* está implementado na linguagem *Java* em sua versão 5.0 [27]. A escolha desta linguagem deve-se principalmente as seguintes caracteríticas:

- permitir integração com as funcionalidades e *APIs* disponibilizadas pelo *framework JADE* (versão 3.4.1) [28];

- permitir integração com o motor de inferência *JESS* [93], versão 7.0, utilizado para a construção do mecanismo de raciocínio - módulo FC - dos agentes. Para um melhor entendimento, o *JESS* é utilizado para construir bases de conhecimento e obter inferências a partir de padrões pré-estabelecidos, sendo especialmente desenvolvido para ser integrado à linguagem *Java*, o que permite a criação de *softwares Java*, inclusive SMA, com capacidade de resolução de problemas usando conhecimento vindo das regras de produção implementadas em *JESS*, ou seja, permitindo que o módulo de negócio (o que fazer) dos agentes seja disjunto do seu módulo de raciocínio (como fazer), fornecendo ao desenvolvedor a facilidade de aprimorar os algoritmos de controle interno dos agentes sem quaisquer modificações em seu módulo de raciocínio.;

- ser a linguagem dos códigos gerados pelo aplicativo *OntologyBeanGenerator* [63, 60, 35], *plugin* utilizado para gerar códigos *Java*, a partir de ontologias construídas na ferramenta *Protégé* [35], que podem ser utilizados por SMA desenvolvidos sob o ambiente *JADE*. *Protégé* é um *software* que fornece um ambiente integrado, apresentando uma *interface* amigável e de simples manipulação, usado por desenvolvedores de sistemas e especialistas em um domínio específico para o desenvolvimento de ontologias [35, 131];

- ser uma linguagem portavél, permitindo que o *BioAgents* seja utilizado em diferentes plataformas.

4.2.2 Descrição das Camadas

Nessa subseção apresentamos a composição e o funcionamento das camadas nas quais o protótipo *BioAgents* está estruturado.

Camada de Interface

A Camada de *Interface* é responsável por receber as solicitações submetidas ao sistema e retornar o resultado do processamento ao usuário, no caso, os biólogos responsáveis pela anotação manual. A solicitação é descrita como uma mensagem (requisição) que é enviada pelo *Agente de Interface* (AI), denominado *StatisticsAnalysisAgent*, ao *Agente Resolvedor de Conflito* (RC), denominado *ConflictResolutionAgent*, no momento da execução do sistema (a chamada ao sistema é realizada através de linha de comando *batch*).

No *BioAgents*, para a apresentação dos resultados, o agente AI recebe o conjunto de sugestões enviadas pelo agente RC e compara-as diretamente com as anotações curadas (anotadas manualmente) pelos biólogos. A Figura 4.3 mostra um *screenshot* de execução do *BioAgents*, onde a tela denominada *Analysis Agent* é uma *interface* simples de inicialização do sistema.

Camada Colaborativa

A Camada Colaborativa é o núcleo do *BioAgents*. Esta camada consiste de um módulo de resolução de conflitos, composto pelo agente RC, e um módulo central organizado conforme uma abordagem *Blackboard*. A estrutura da Camada Colaborativa é composta pelo agente RC, *Agentes Gerentes* (GR), *Agentes Analisadores* (ANL), módulo FC e um componente BB.

O agente RC, de acordo com o observado na Camada de *Interface*, é responsável por receber a solicitação de anotação enviada pelo agente AI, bem como enviar à Camada de *Interface* o conjunto de sugestões fornecido pelos agentes GR. Após receber a mensagem solicitando a anotação, o agente RC envia uma mensagem aos agentes GR com o objetivo de ativá-los. É importante enfatizar que nesta implementação do *BioAgents*, o agente RC não possui o módulo de resolução de conflitos implementado.

Os agentes GR, denominados *BlastManagerAgent* e *FastaManagerAgent*, são os responsáveis por comandar o ciclo de execução dos agentes ANL, conforme

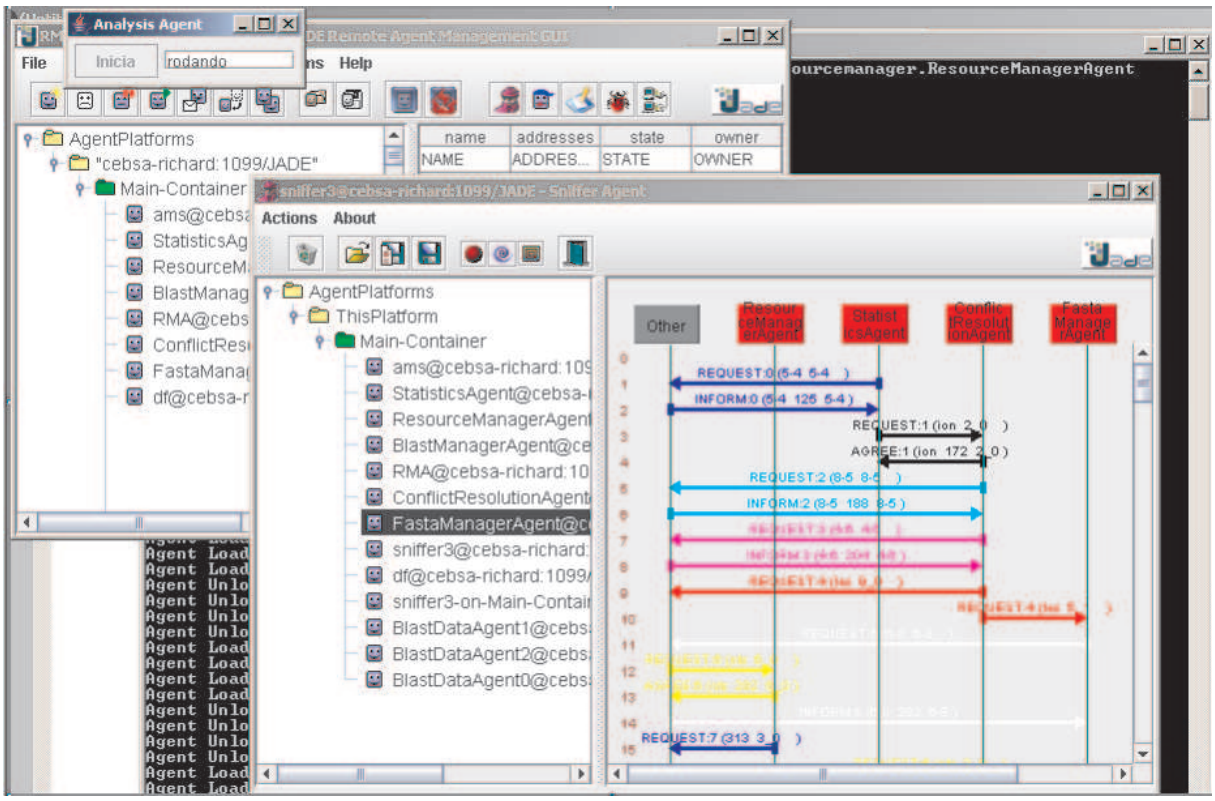


Figura 4.3: *Screenshot* da tela de execução e do *sniffer* (módulo disponibilizado pelo *framework JADE*) mostrando as mensagens trocadas entre agentes do protótipo *BioAgents*.

descrito na Seção 4.2.1. Os agentes GR também centralizam e analisam as sugestões enviadas pelos agentes ANL. Os agentes GR são ativados pelo agente RC de acordo com as suas especialidades e as necessidades do projeto de seqüenciamento. Estes agentes, após as análises realizadas pelos agentes ANL sobre as bases de dados disponíveis na Camada Física, consolidam as sugestões fornecidas pelos agentes ANL através de regras de produção próprias, implementadas nos módulos FC Gerente (Figura 4.2). Na Figura 4.4 é apresentado um fluxo genérico de comunicação entre os agentes GR e os agentes ANL, representado por meio de etapas que estão numeradas de 1 a 8. Estas etapas são descritas como:

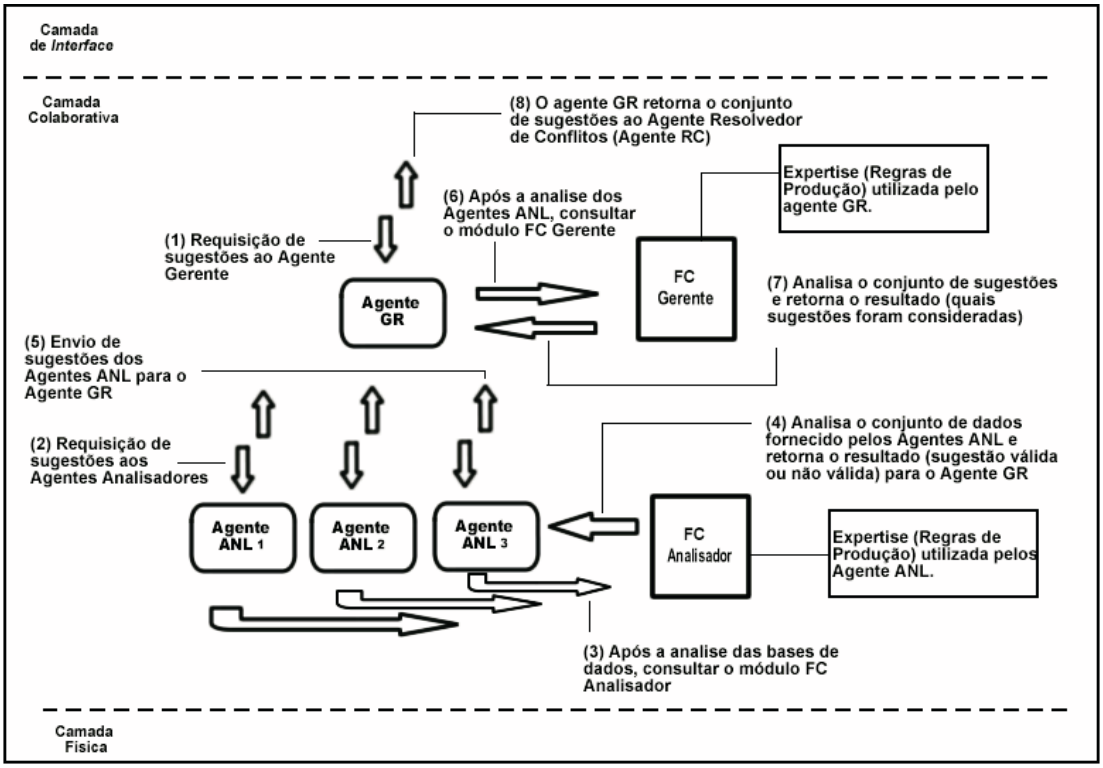


Figura 4.4: Fluxo de comunicação entre os agentes GR e agentes ANL.

1. o agente RC envia ao agente GR uma mensagem solicitando sugestão de anotação;

2. o agente GR então envia mensagens aos agentes ANL solicitando uma análise destes agentes às bases de dados contidas na Camada Física;

3. após os agentes ANL terem recuperado os dados da Camada Física, estes agentes interagem com o componente FC Analisador. Esta interação tem por objetivo a análise dos dados recuperados da Camada Física, por cada agente ANL, pelas regras de produção implementadas neste componente.

4. após a análise dos dados, o módulo FC Analisador retorna aos agentes ANL, obedecendo a solicitação de cada um, a resposta em relação à avaliação do conjunto de regras de produção implementadas no módulo FC;

5. os agentes ANL então enviam suas sugestões ao agente GR;

6. o agente GR recebe as solicitações do agentes ANL e, de forma análoga a etapa 3, solicita ao módulo FC Gerente a análise das sugestões enviadas pelos agentes ANL;

7. de forma análoga a etapa 4, o módulo FC Gerente retorna ao agente GR a resposta acerca de sua análise, identificando quais sugestões enviadas pelos agente ANL serão repassadas ao agente RC;

8. O agente GR consolida os resultados e envia suas sugestões ao agente RC.

As mensagens trocadas entre os agentes obedecem a estrutura da linguagem *FIPA-ACL* [31], conforme apresentado na Seção 3.3.1. Para exemplicar, o código abaixo apresenta a estrutura de uma mensagem relacionada ao pedido de um agente GR para um agente ANL a fim de que seja executada a análise do arquivo de saída `blastXnr`, o qual contém os resultados da análise feita pelo *software BLASTX* sobre a base de proteínas *nr*.

```
(REQUEST
:sender (agent-identifier
   :name BlastManagerAgent )
:receiver (set (agent-identifier
   :name BlastAnalysisAgent0 ) )
:content "((action (agent-identifier
   :name BlastManagerAgent )
       (RequestFile
       :FilePath c:\anotManualPb\Contig1013\Contig1013.blastXnr
       :FileType 0
       :FileLocation \"\")))"
:ontology  CommunicationOntology)
```

Neste código alguns detalhes e campos foram omitidos por serem utilizados apenas para controle interno do *framework JADE*. O `REQUEST` significa o tipo de ato comunicativo que esta sendo utilizado, neste caso, uma solicitação. O *BioAgents* faz uso de quatro atos comunicativos, a saber: *agree*, *request*, *inform* e *not_understood* conforme definidos na Seção 3.3.1. O campo `sender` indica o agente que está originando a solicitação, no caso o agente GR `BlastManagerAgent`. O campo `receiver` refere-se ao receptor responsável por executar a solicitação, identificado pelo agente ANL `BlastAnalysisAgent0`. Em relação ao campo `content`, este traz a ação (`action`) a ser executada. Os detalhes do conteúdo das sugestões e os conceitos que as representam serão apresentados adiante ao descrevermos os vocabulários utilizados pelo *BioAgents*. Por fim, o campo `ontology` indica o vocabulário que está sendo utilizado (`CommunicationOntology`) entre o agente solicitante (agente GR) e o agente receptor (agente ANL) a fim de que a comunicação seja estabelecida. Este vocabulário é necessário para que os agentes entendam da mesma forma o conteúdo de uma solicitação ou de uma resposta.

Os agentes ANL, denominados *BlastAnalysisAgent* e *FastaAnalysisAgent*, constituem o último nível da Camada Colaborativa, sendo os responsáveis por enviar sugestões individuais aos agentes GR. As sugestões fornecidas pelos agentes

ANL, assim como nos agentes GR, são viabilizadas por regras de produção implementadas nos módulos FC específicos a estes agentes, conforme apresentado na Figura 4.2. O conteúdo das sugestões retornadas pelos agentes ANL e GR é visualizado no diagrama de classes apresentado na Figura 4.5.

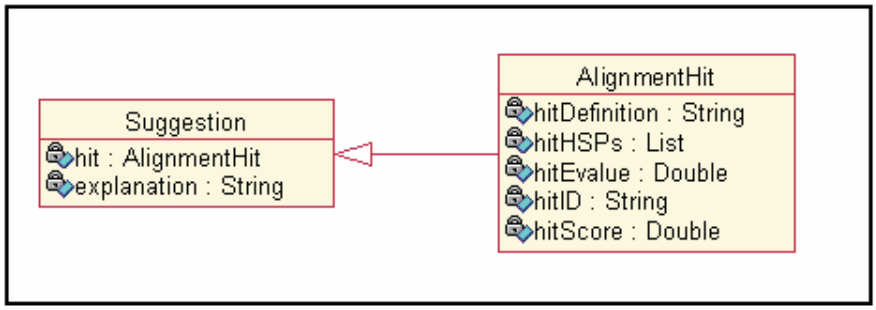


Figura 4.5: Conteúdo e estrutura das sugestões enviadas pelos agentes ANL aos agentes GR.

Na Figura 4.5 podemos observar que as sugestões são compostas das classes *Suggestion* (classe mais geral) e *AlignmentHit* (classe mais específica). Estas classes foram criadas a partir do vocabulário declarado no componente BB, construído com o uso da ferramenta *Protégé*. Dentre as informações fornecidas pelas saídas dos *softwares BLAST* e *FASTA*, estamos considerando para as sugestões os atributos relacionados ao *e-value*, *score*, o identificador do alinhamento (*hitID*) e o nome da seqüência (*hitDefinition*) referente ao alinhamento identificado. Como exemplo dos conteúdos atribuídos a estes atributos, a Figura 4.6 apresenta o cabeçalho de um alinhamento obtido na tarefa de anotação automática do Projeto Genoma Guaraná, com a execução do *BLAST* (programa *BLASTX*) sobre a base *nr*. Na Figura 4.6 o *e-value* é representado pelo valor `2e-33`, o *score* é identificado pelo valor `360`, o *hitID* é representado pelo identificador `gb|AAM64995.1|` e o *hitDefinition* é representado pelo nome `unknown [Arabidopsis thaliana]`. Os atributos *e-value* e *score* estão descritos na Seção 2.5 e são os parâmetros de avaliação utilizados pelas regras de produção implementadas nos módulos FC.

```
>gb|AAM64995.1| unknown [Arabidopsis thaliana]
          Length = 240

 Score =  143 bits (360), Expect = 2e-33
 Identities = 75/129 (58%), Positives = 82/129 (63%), Gaps = 6/129 (4%)
 Frame = +3
```

Figura 4.6: Exemplo de um alinhamento obtido com a execução do programa *BLASTX*.

A quantidade de agentes ANL criadas pelos agentes GR é definida dinamicamente de acordo com a quantidade de bases de dados disponíveis na Camada Física, ou seja, se o projeto possui três bases de dados então os agentes GR deverão alocar três agentes ANL para a execução das análises. À medida que os agentes ANL finalizam suas tarefas, os agentes GR os encerram. Esta ação permite ao *BioAgents* não utilizar recursos de memória física de forma desnecessária.

A interação entre os agentes ANL e as bases de dados é realizada por meio de *parsers* específicos aos tipos de dados utilizados pelo sistema. Estes *parsers* são utilizados para fornecer aos agentes ANL a estrutura de dados - uma lista contendo as sugestões destes agentes - necessária para a análise dos agentes GR. As informações requeridas pelos agentes ANL são acessadas e recuperadas do componente BB. Os agentes GR também se comunicam com o BB, pois estes devem informar aos agentes ANL, no momento de sua ativação, qual o vocabulário que será utilizado para a comunicação entre estes agentes e os agentes GR, conforme visualizado na Figura 4.2. Na atual implementação do protótipo não estão sendo tratadas as falhas de comunicação.

Na Figura 4.7 é apresentado o diagrama de seqüência correspondente a comunicação entre a *Camadas de Interface*, identificada pelo agente AI (*StatisticsAnalysisAgent*), e a Camada Colaborativa, identificada pelo agente RC (*ConflictResolutionAgent*), o agente GR *BLAST* (*BlastManagerAgent*), um agente ANL *BLAST* (*BlastAnalysisAgent*) e as FC no nível dos agentes GR (*SuggestionsAnalyzer*) e dos agentes ANL (*FileAnalyzer*). O "X"visualizado logo abaixo do agente ANL representa o término do ciclo de vida deste agente, e o *loop* (laço) encontrado nos componentes FC *FileAnalyzer* e *SuggestionsAnalyzer*, indica que a mensagem de retorno enviada por estes dois componentes será ativada somente após o término das análises executadas pelos agentes GR e ANL.

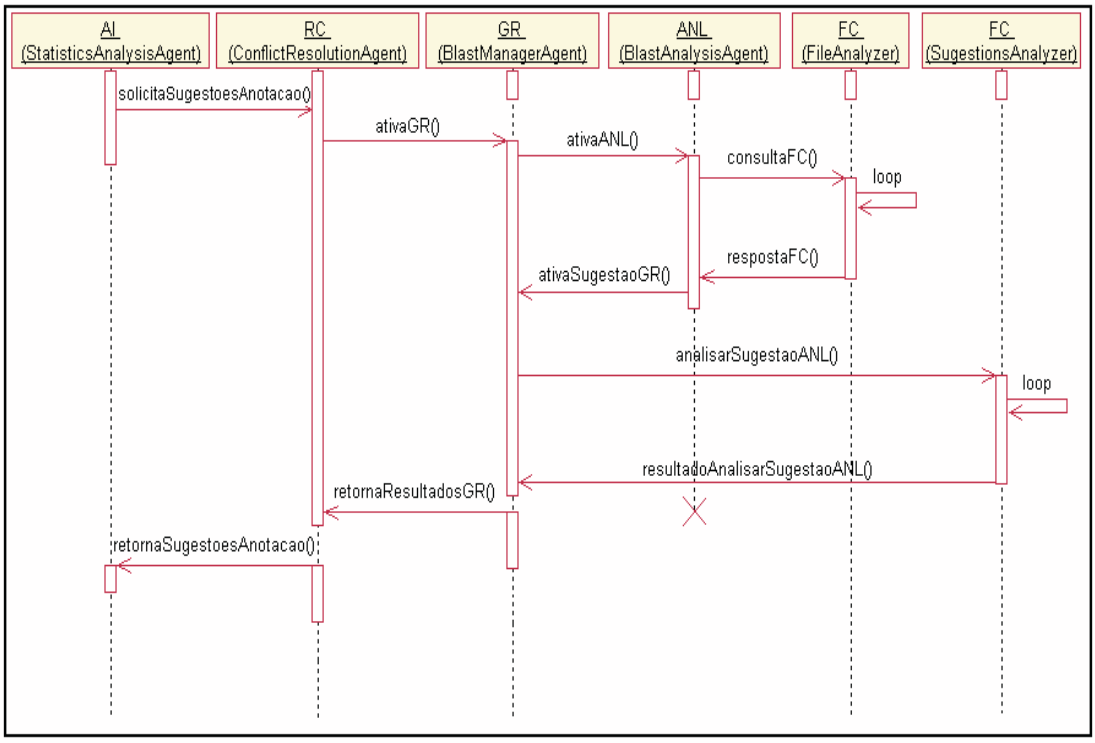


Figura 4.7: Ciclo de vida e o fluxo de troca de mensagens entre os agentes do protótipo *BioAgents*, considerando o exemplo de sugestões a partir da análise de resultados do *software BLAST*.

Em relação aos componentes FC, como descrito anteriormente, estes representam os módulos de raciocínio do *BioAgents*, os quais incluem a *expertise* dos

agentes representadas através de regras de produção implementadas no motor de inferência *JESS*. A nível dos agentes GR, as FC são denominadas como *BlastSugestionsAnalyzer* e *FastaSugestionsAnalyzer*, e no caso dos agentes ANL, são denominadas como *BlastFileAnalyzer* e *FastaFileAnalyzer*. A comunicação entre os agentes e o *JESS* é estabelecida por meio de métodos disponibilizados por *APIs* (bibliotecas) *JESS*.

Os componentes FC são declarados no nível dos agentes GR com o objetivo de consolidar e analisar, por meio das regras de produção implementadas, quais sugestões fornecidas pelos agentes ANL serão repassadas ao agente RC, para que após o envio de todas as sugestões dos agentes GR as sugestões sejam encaminhadas a Camada de *Interface*. A estrutura, sintaxe e descrição das regras utilizadas pelo *BioAgents* para fornecer as sugestões de anotação manual são apresentadas no Capítulo 5.

Finalizando a descrição da Camada Colaborativa, temos o componente BB. Este componente possui vocabulários, ou seja, um conjunto de conceitos representados por classes que especificam a comunicação entre os agentes. Os vocabulários foram desenvolvidos na ferramenta *Protégé* [35]. É importante enfatizar que os agentes GR e ANL não atualizam automaticamente o BB, isto é, estes agentes apenas consultam e recuperam as informações necessárias contidas neste componente. Esta consulta e recuperação de informações é viabilizada por meio de métodos inerentes a cada classe escrita nos vocabulários. Estes métodos são gerados automaticamente pela ferramenta *OntologyBeanGenerator* [63, 60, 35].

O componente BB, conforme descrito anteriormente, contém apenas os vocabulários utilizados para a comunicação e representação de informações biológicas encontradas nas bases de dados utilizadas pelo *BioAgents*. Estes vocabulários são denominados *BioOntology* e *CommunicationOntology*.

O vocabulário *BioOntology* é utilizado para estruturar classes que representem os diversos parâmetros que são encontrados nos arquivos de saída dos *softwares* utilizados na tarefa de anotação automática. A classe *AlignmentHit*, apresentada na Figura 4.5, é um exemplo de classe pertencente ao vocabulário *BioOntology*.

A Figura 4.8 mostra a estrutura do vocabulário *BioOntology*, onde podemos observar classes relacionadas aos arquivos de saída (`BlastOutput` e `FastaOutput`) dos *softwares BLAST* e *FASTA*.

No que se refere ao vocabulário *CommunicationOntology*, este é responsável por estruturar classes inerentes a comunicação entre os agentes, bem como generalizar as classes que compõem as sugestões fornecidas pelos agentes ANL. Esta generalização pode ser observada na Figura 4.5, onde a classe *Suggestion*, pertencente ao vocabulário *CommunicationOntology*, generaliza uma determinada sugestão fornecida por um agente ANL, através da classe *AlignmentHit* pertencente ao vocabulário *BioOntology*.

A Figura 4.9 apresenta a estrutura de classes descritas no vocabulário *CommunicationOntology*, necessárias à comunicação entre os agentes que compõem a arquitetura do protótipo *BioAgents*. Na figura, além da classe *Suggestion* apresentada anteriormente, podemos observar as classes *Suggests*, *RequestConfig*, com as extensões *RequestAlgorithm* e *RequestFile*, *GroupID* e a *interface AgentAction*, pertencente a uma das *APIs* disponibilizadas pelo *framework JADE*, com as realizações *CreateAgent*, *KillAgent* e *RequestSuggestion*.

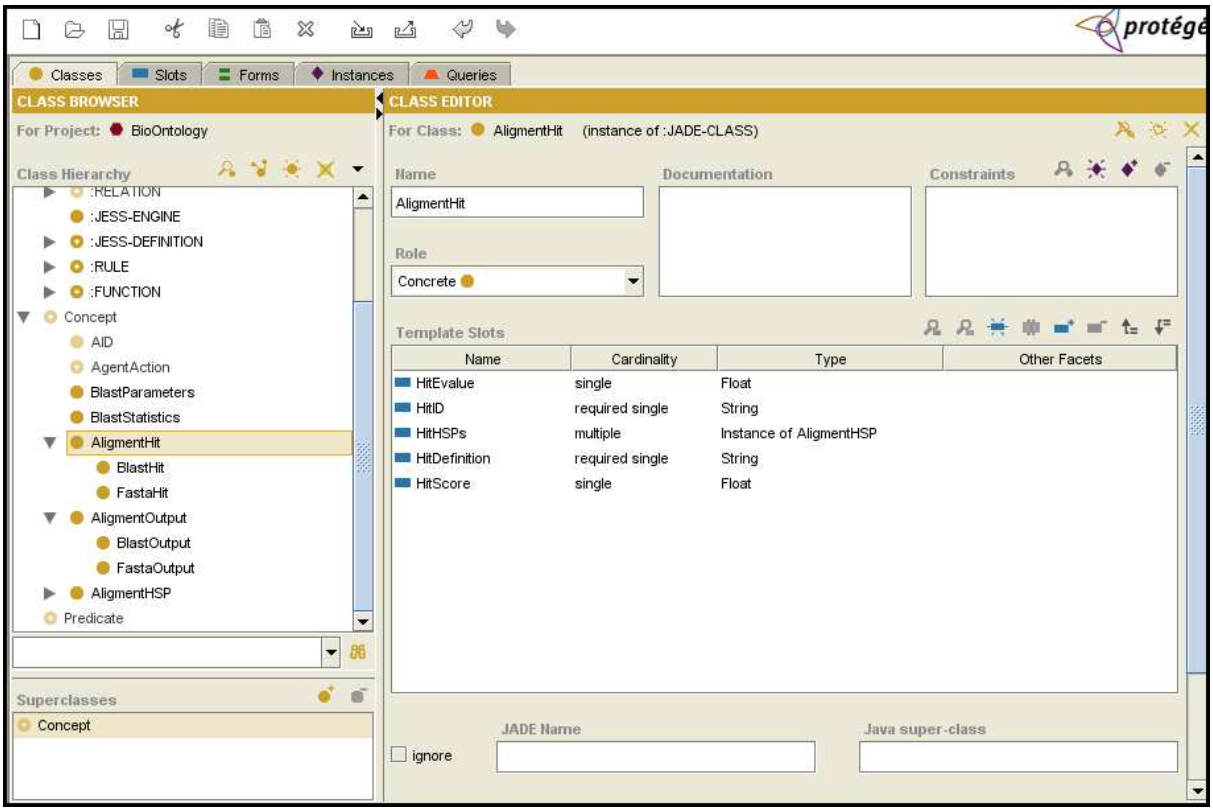


Figura 4.8: Estrutura do vocabulário *BioOntology*.

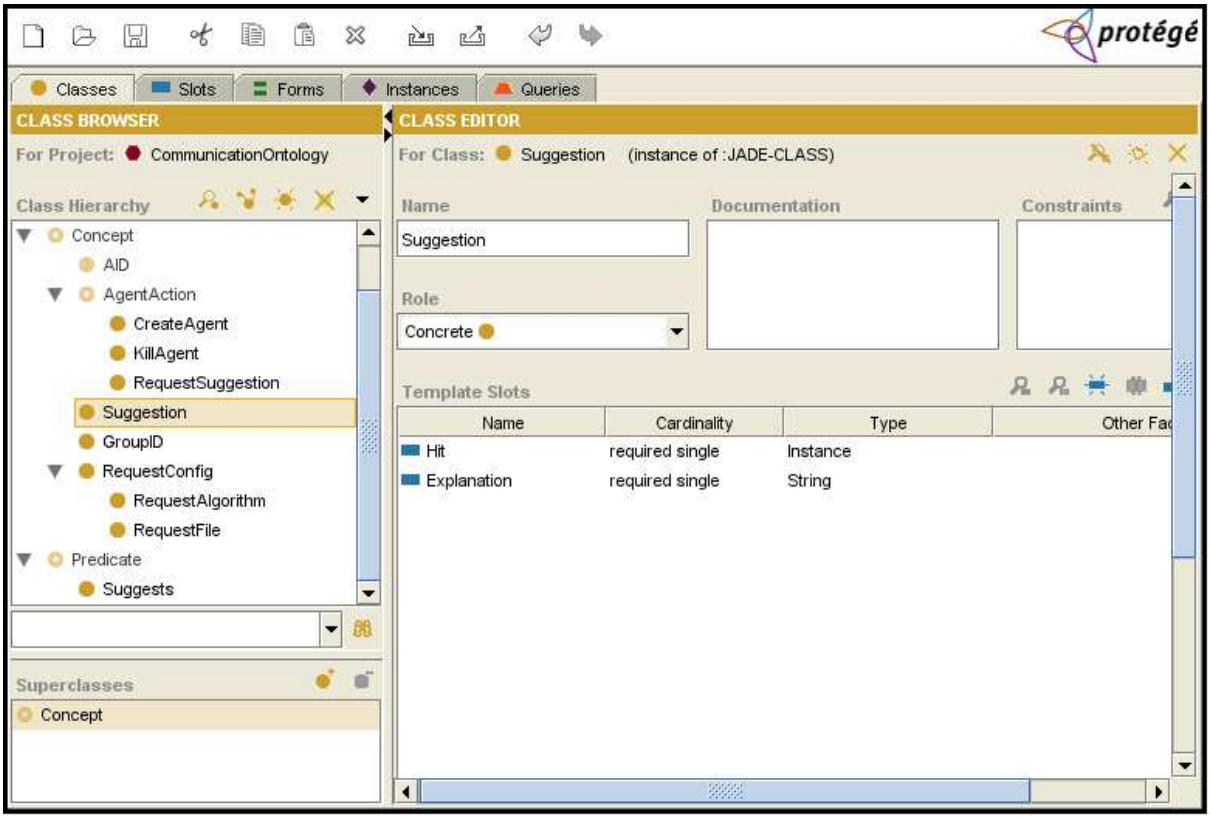


Figura 4.9: Estrutura do vocabulário *CommunicationOntology*.

A classe *Suggests* é utilizada para relacionar um conjunto de sugestões ao agente que as originou. A Figura 4.10 apresenta o relacionamento entre as classes que compõem o conjunto de sugestões fornecidos pelo *BioAgents*.

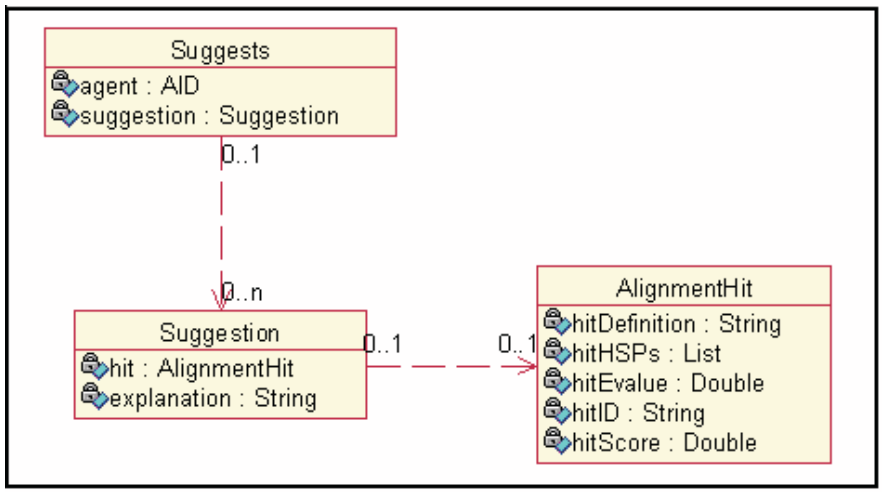


Figura 4.10: Relacionamento entre as classes que compõem as sugestões fornecidas pelo *BioAgents*.

A classe *RequestConfig*, juntamente com suas extensões, refere-se a forma na qual as bases de dados estão armazenadas na Camada Física. O *BioAgents* foi desenvolvido para permitir aos usuários trabalharem com duas formas de armazenamento: arquivos texto ou banco de dados relacional. Isto significa que o *BioAgents* pode analisar bases de dados formadas apenas por arquivos do tipo texto quanto bases de dados gerenciadas por um banco de dados relacional (por exemplo, o banco *PostgreSQL* [36]). Na Figura 4.11 podemos observar o relacionamento entre as classes *RequestConfig*, *RequestAlgorithm* e *RequestFile*. O atributo *requestType* informa ao *BioAgents* o tipo de armazenamento que será utilizado.

A importância da classe *GroupID* consiste em permitir a identificação de quais grupos, por exemplo, *contigs* ou *singlets* (Seção 2.4.1), estão relacionadas as sugestões fornecidas pelo *BioAgents*.

Por fim, na Figura 4.9, pode-se observar as classes *CreateAgent*, *KillAgent* e *RequestSuggestion*. As classes *CreateAgent* e *KillAgent* são especialmente utilizadas pelos agentes GR para criarem e finalizarem agentes ANL. Estas ações são motivadas pelo fato de permitir ao *BioAgents* não consumir recursos de máquina quando há agentes ANL inativos, isto é, que não estejam executando alguma tarefa. Em relação a classe *RequestSuggestion*, sua estrutura permite aos agentes GR recuperar qual agente reportou uma determinada sugestão, a que *GroupID* a sugestão está vinculada, bem como a origem (fonte de dados na Camada Física) da sugestão.

Camada Física

A Camada Física consiste das bases de dados formadas pelos resultados da análise

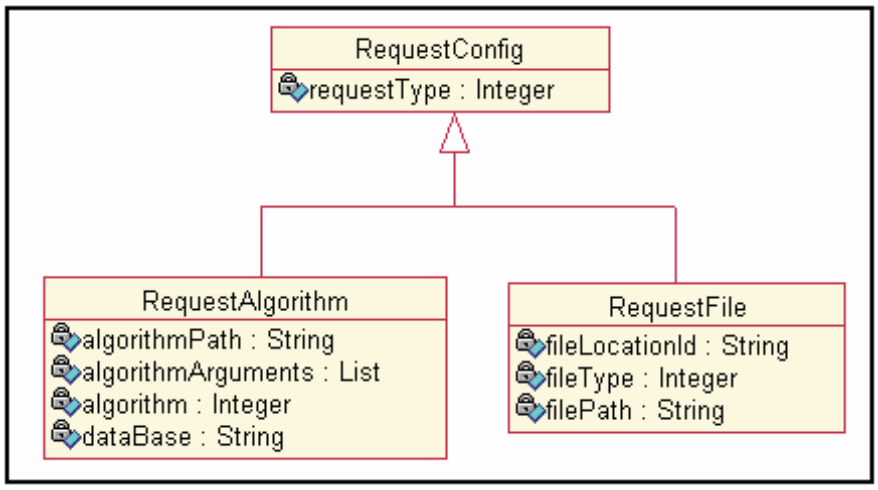


Figura 4.11: Diagrama de classes apresentando a relação entre as classes *RequestConfig*, *RequestAlgorithm* e *RequestFile*.

dos *softwares* utilizados na tarefa de anotação automática, podendo estar armazenadas tanto em formato de arquivos texto como em bancos de dados relacionais. Estas bases de dados fornecem os dados necessários a análise dos agentes ANL.

É importante enfatizar que nesta primeira implementação do protótipo, as bases de dados utilizadas para estudo de caso são compostas apenas dos arquivos de saída com os resultados das comparações efetuadas pelos *softwares BLAST* e *FASTA* sobre as bases *nr*, *xyva (COG)*, *GO*, *KOG*, *SwissProt* e as bases dos fungos *Sacchoromyces cereviseae* (SC) e *Schizosaccharomyces pombee* (SP), conforme apresentado na Figura 4.2.

Nesse capítulo abordamos as características da arquitetura proposta e consequente implementação do protótipo *BioAgents*. Considerando que o escopo da proposta inicial era amplo, não implementamos os módulos relacionados a resolução de conflitos, desenvolvimento de técnicas de mineração de dados e algoritmos de aprendizado de máquina. No entanto, apresentamos de forma detalhada o projeto de implementação do *BioAgents*, o qual foi interamente desenvolvido na linguagem *Java*, visando a integração com o *framework JADE*. Utilizamos e descrevemos o motor de inferência *JESS* para o desenvolvimento das FC. O *software Protégé* foi utilizado no desenvolvimento dos vocabulários relacionados ao componente BB, denominados *BioOntology* e *CommunicationOntology*. No Capítulo 5 será apresentado a descrição das regras implementadas nos módulos FC, a descrição dos estudos de caso utilizados na validação do *BioAgents*, bem como estudos comparativos dos trabalhos correlatos.

Capítulo 5

Estudo de Caso e Discussão dos Resultados

O objetivo desse capítulo é apresentar e comentar os resultados obtidos utilizando-se a estrutura do protótipo *BioAgents* sobre os dados disponibilizados pelos projetos utilizados no estudo de caso desta dissertação - projetos genoma Pb e Guaraná.

Considerações acerca da organização das bases de dados e a tarefa de anotação manual executada nos projetos relacionados como estudo de caso são descritas na Seção 5.1. A descrição e sintaxe das regras de produção implementadas para as FC do *BioAgents* são apresentadas na Seção 5.2. A apresentação e comentários dos resultados obtidos são descritos na Seção 5.3. Alguns trabalhos correlatos ao apresentado nesta dissertação são comentados na Seção 5.4.

5.1 Anotação Manual nos Projetos Genoma Pb e Guaraná

Para auxiliar a adequada interpretação dos resultados alcançados neste trabalho é importante observar o contexto aplicado aos dados fornecidos pelos projetos genoma Pb e Guaraná, bem como entender a execução da anotação manual nestes projetos. Esta observação deve-se ao fato de que o entendimento das anotações manuais, em qualquer projeto de seqüenciamento, é diretamente relacionado aos tipos e a natureza dos dados considerados na execução da anotação automática.

Conforme descrito na Seção 4.2, as bases de dados utilizadas pelo *BioAgents* são formadas por arquivos de saída com os resultados das anotações automáticas. Sendo assim, estas bases de dados estão divididas da seguinte maneira:

- Projeto Genoma Pb: consiste dos arquivos de saída com os resultados originais obtidos com a execução do *software BLASTX* sobre as bases de dados *nr*, *xyva (COG)* e *GO*, e com a execução do *software FASTA* sobre as bases *Saccharomyces cereviseae* (SC) e *Schizosaccharomyces pombee* (SP).

- Projeto Genoma Guaraná: consiste dos arquivos de saída com os resultados originais obtidos com a execução do *software BLASTX* sobre as bases de dados *nr*, *KOG*, *GO* e *SwissProt*.

Os resultados das anotações automáticas destes projetos possuem naturezas semelhantes, a saber:

- as seqüências de ambos os projetos são genes expressos (*EST*) - genes que codificam produtos (proteínas ou *RNAs*) - as quais foram divididas em grupos de *contigs* e *singlets*;

- O programa *BLAST* utilizado por ambos os projetos foi o *BLASTX*. Relembrando, este programa tem como entrada uma seqüência de genes (nucleotídeos) que são traduzidos para uma seqüência de proteínas. Estas seqüências são analisadas com bases de dados de proteínas, conforme descrito na Seção 2.5.

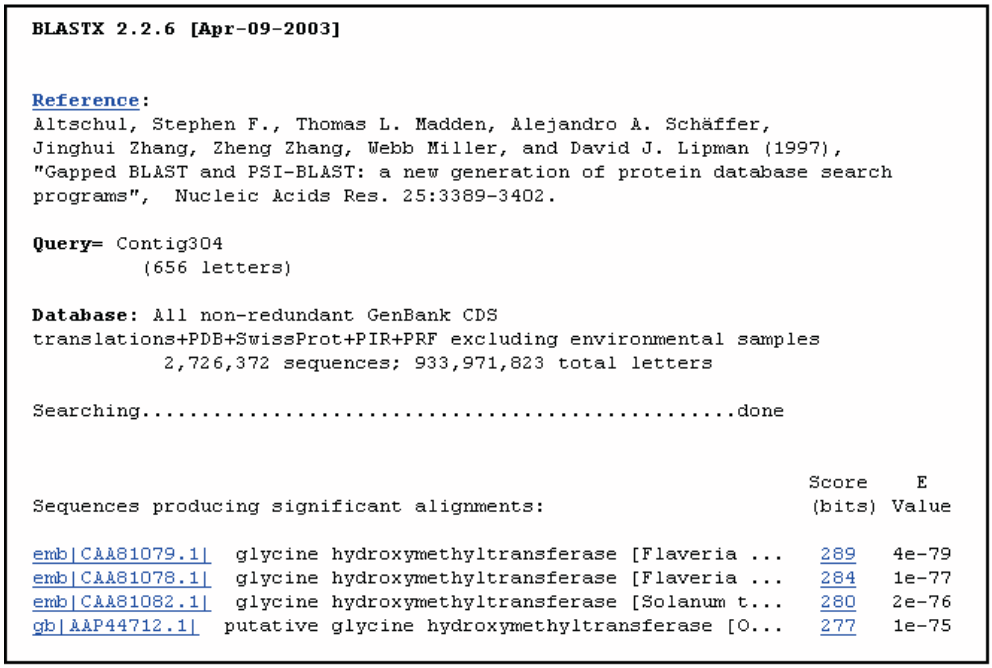
Na Figura 5.1 é apresentado um trecho de um dos arquivos de saída utilizados pelo *BioAgents* para fornecer as sugestões de anotações manuais. De acordo com a estrutura descrita acima, as bases de dados que compõem a Camada Física da arquitetura do protótipo *BioAgents*, conforme apresentada na Seção 4.2.1, devem ser interpretadas da seguinte maneira:

> A base *nr* significa a base de dados com os arquivos de saída contendo os resultados originais obtidos com a execução do programa *BLASTX* sobre a base de proteínas *nr*. A base SC significa a base de dados com os arquivos de saída contendo os resultados originais obtidos com a execução do programa *FASTA* sobre a base do fungo *Saccharomyces cereviseae*. Para as bases *xyva (COG)*, *GO*, *KOG*, *Swiss-Prot* e SP a interpretação é análoga.

Em relação às anotações manuais, os biólogos utilizaram como ferramenta para a execução desta atividade as páginas *Web* pertencentes aos módulos de anotação manual disponibilizados pelos sistemas do tipo *pipeline* desenvolvidos pelo Laboratório de Bioinformática da UnB. Estas páginas *Web* são apresentadas nas Figuras 5.2 e 5.3 e possuem acesso restrito aos pesquisadores que participam de projetos suportados por este Laboratório de Bioinformática.

Como podemos observar nas Figuras 5.2 e 5.3, os projetos genoma Pb e Guaraná possuem campos idênticos para a anotação manual, devido a natureza semelhante destes projetos com diferenças apenas no grupo ortólogo utilizado: *COG* no Projeto Genoma Pb e *KOG* no Projeto Genoma Guaraná. Estes campos são denominados *Nome do Produto*, *Categoria e Nome do COG* (Projeto Genoma Pb) e *Categoria e Nome do KOG* (Projeto Genoma Guaraná), Número *Enzyme Commission* (Número *EC*)[1] *GO*, a identificação *KEGG*, *Categorização dos Genes* e *Nome do Gene*. A descrição de cada campo não é detalhada, pois não é objetivo deste trabalho apresentar a importância e os detalhes técnicos de todos os campos anotados pelos biólogos na tarefa de anotação manual.

[1]Trata-se de um número atribuído para o tipo de enzima (proteína que estimula uma reação bioquímica) de acordo com um esquema de nomenclatura desenvolvido pela Comissão de Enzimas do Comitê de Nomenclaturas da União Internacional de Bioquímica e Biologia Molecular [9].



```
BLASTX 2.2.6 [Apr-09-2003]


Reference:
Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schäffer,
Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
"Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs",  Nucleic Acids Res. 25:3389-3402.

Query= Contig304
         (656 letters)

Database: All non-redundant GenBank CDS
translations+PDB+SwissProt+PIR+PRF excluding environmental samples
           2,726,372 sequences; 933,971,823 total letters

Searching.................................................done


                                                                  Score    E
Sequences producing significant alignments:                       (bits) Value

emb|CAA81079.1|  glycine hydroxymethyltransferase [Flaveria ...   289    4e-79
emb|CAA81078.1|  glycine hydroxymethyltransferase [Flaveria ...   284    1e-77
emb|CAA81082.1|  glycine hydroxymethyltransferase [Solanum t...   280    2e-76
gb|AAP44712.1|  putative glycine hydroxymethyltransferase [O...   277    1e-75
```

Figura 5.1: Arquivo de saída (em formato *html*), armazenado em um banco *PostgreSQL*, contendo os resultados originais obtidos com a execução do programa *BLASTX* sobre a base de proteínas *nr*.

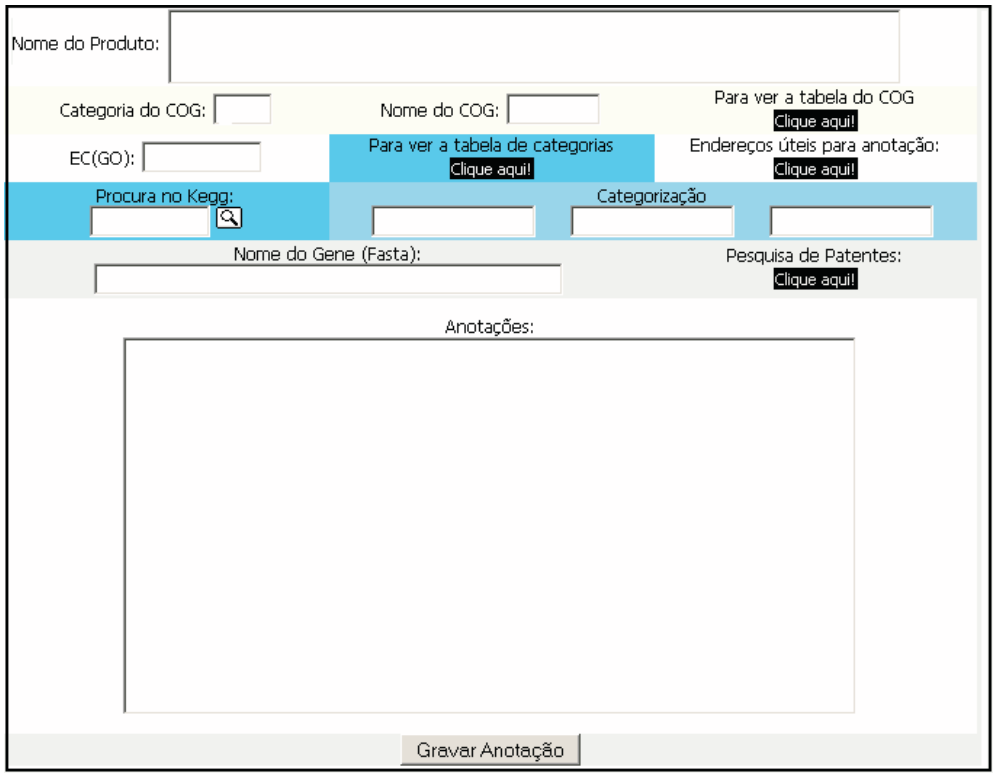


Figura 5.2: Página *Web* utilizada para a anotação manual do Projeto Genoma Pb.

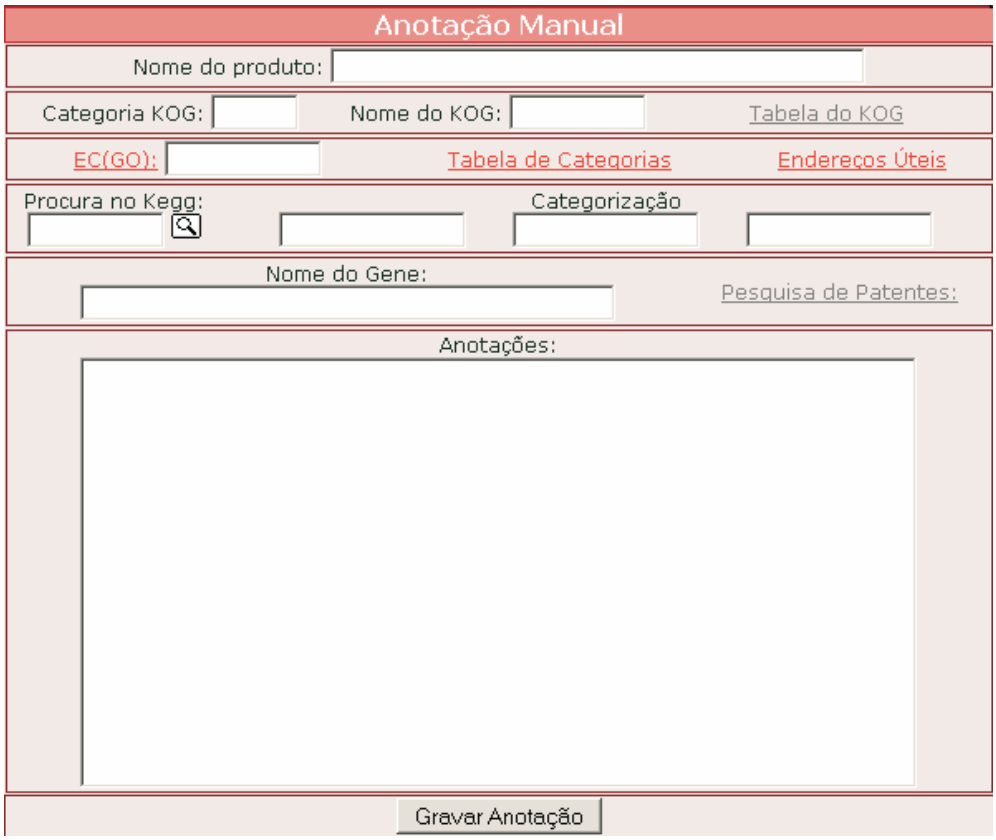


Figura 5.3: Página *Web* utilizada para a anotação manual do Projeto Genoma Guaraná.

5.2 Regras utilizadas

Na Seção 4.2.2 foram apresentados os componentes FC. Estes componentes representam os mecanismos de raciocínio que incluem a *expertise* do protótipo *BioAgents*. A *expertise* é representada por regras de produção implementadas no motor de inferência *JESS* [93].

Podemos ressaltar que estas regras de produção foram definidas a partir de reuniões com biólogos e têm como objetivo representar o conhecimento explícito utilizado por estes especialistas na execução da tarefa de anotação manual dos projetos genoma Pb e Guaraná [107, 106]. As regras são utilizadas para analisar as bases de dados que compõem a Camada Física da arquitetura do protótipo *BioAgents*, a partir dos parâmetros *e-value* e *score*. O significado e a importância da avaliação destes parâmetros foram apresentados na Seção 2.5. No entanto, estes parâmetros podem ser entendidos como:

- *e-value*: corresponde a possibilidade de um determinado alinhamento acontecer por acaso, sendo que quanto menor for o seu valor maior são as chances do alinhamento reportar uma similaridade real;
- *score*: é a medida que informa o quanto duas seqüências, de um determinado alinhamento, são similares.

No protótipo *BioAgents* implementado durante a execução deste trabalho utilizamos três regras de produção básicas, as quais capturam o seguinte conhecimento biológico na tarefa de anotação manual:

1. Verificar a existência de alinhamentos cujo *e-value* seja menor ou igual a 10^{-5} (faixa de aceitação estabelecida pelos biólogos nos projetos genoma Pb e Guaraná);
2. Dentre os alinhamentos que atendem à restrição anterior, selecionar o menor *e-value*;
3. Caso existam dois *e-values* iguais, selecionar o de maior *score*.

O extrato de código abaixo apresenta a regra 1 descrita acima. A sintaxe da linguagem representada pelo código é própria do sistema *JESS*, possuindo uma grande semelhança à da linguagem *List Processing* (*LISP*), bastante conhecida por desenvolvedores de sistemas inteligentes [93].

```
;; global variable that represents the e-value
(defglobal ?*maxEvalue* = 1.0E-5)

(defrule Exists_Evalue_Below_Limit
       (exists (BlastHit
                (HitEvalue ?evalue&:(<= ?evalue ?*maxEvalue*))
    ))
    =>
    (focus GoodEvalueAnalysis)
    (run)
)
```

No código observamos alguns comentários iniciados pelo caractere `;` e a definição de uma variável global denominada `maxEvalue` que serve para atribuir o valor `1.0E-5` (10^{-5}), o qual refere-se ao valor do *e-value* (conforme regra 1). A palavra `defrule` define uma regra que é disparada quando os agentes fazem uma chamada ao componente FC, por meio de comandos *JESS*. Na regra denominada `Exists_Evalue_Below_Limit`, conforme observada no código, caso houver *hits* com *e-values* menores que o valor atribuído a váriável `maxEvalue`, o módulo `GoodEvalueAnalysis` será executado. Este módulo, codifica as regras 2 e 3 descritas anteriormente. O código *JESS* completo que representa as regras é apresentado no Apêndice A.

Para um melhor entendimento do fluxo de execução das regras utilizadas, o diagrama de atividades da Figura 5.4 apresenta o conjunto de instruções relacionados às regras 1, 2 e 3 descritas.

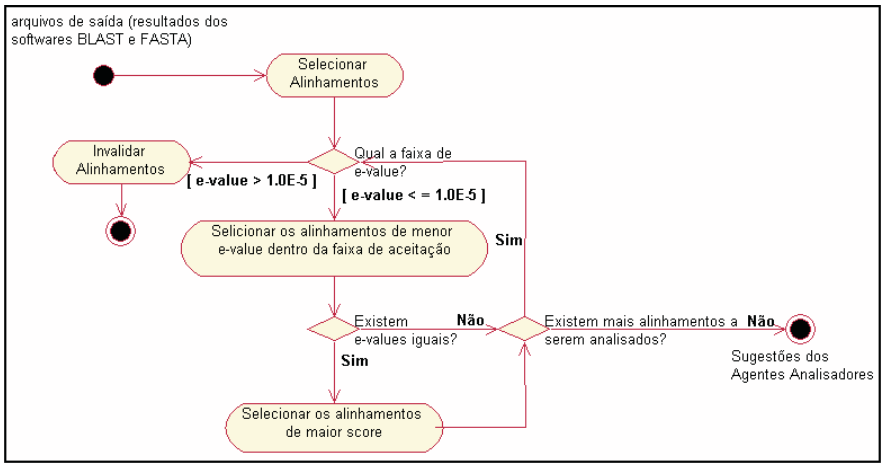


Figura 5.4: Diagrama de atividades referente ao fluxo de instruções das regras de produção.

O fluxo de instruções apresentado na Figura 5.4 representa a *expertise* utilizada pelos agentes ANL. No caso dos agentes GR, o fluxo é análogo. No entanto, existe a diferença de que os agentes GR não verificam um *e-value* com valor específico, mas decidem pela sugestão com menor *e-value* dentre as enviadas pelos agentes ANL.

É importante ressaltarmos que apesar da *expertise* do protótipo *BioAgents* estar representada apenas por três regras conforme descrito anteriormente, outras regras foram previamente definidas com os biólogos do Laboratório de Bioinformática da UnB. No entanto, serão implementadas em trabalhos futuros. Como exemplo, podemos citar uma regra para verificação dos bancos relacionados aos resultados reportados pelo *BLAST* e o *FASTA*. Atribuindo uma escala de confiabilidade a esta regra, por exemplo, a base *Swiss-Prot* deverá receber o maior peso devido a qualidade das anotações armazenadas e ao seus baixos níveis de redundância [18].

5.3 Resultados obtidos

Nessa seção apresentamos algumas considerações importantes para o entendimento do estudo de caso, bem como os resultados alcançados com o desenvolvimento do protótipo *BioAgents*.

Visando os objetivos comentados na Seção 1.3, as anotações manuais sugeridas pelo sistema consistiram da anotação do campo *Nome do Produto* pelo *BioAgents*, conforme visualizado nas Figuras 5.2 e 5.3. Este campo é utilizado para identificar os produtos (proteínas ou *RNA*) relacionados aos genes expressos (*EST*) dos projetos genoma Pb e Guaraná.

Um outro detalhe importante que será considerado na apresentação dos resultados, é o fato dos biólogos, originalmente, terem anotado o campo *Nome do Produto* com base em três classificações, a saber:

- anotado com sucesso, ou seja, a anotação automática reportou bons alinhamentos que deram subsídios a anotação do campo;
- não-conclusivo (nc), ou seja, a anotação automática reportou alinhamentos com valores de *e-values* maiores que 10^{-5} e baixos *scores*;
- não-identificado (nid), ou seja, a anotação automática não reportou alinhamentos.

Observamos que as anotações manuais existentes nos projetos genoma Pb e Guaraná utilizaram as três classificações de anotações acima.

Por último, os testes executados com o protótipo *BioAgents* foram realizados em dois estudos de caso. O primeiro consistiu em obter as sugestões de anotação manual utilizando-se a base de dados do Projeto Genoma Pb - Seção 5.3.1. No segundo utilizamos a bases de dados do Projeto Genoma Guaraná - Seção 5.3.2. A composição destas bases de dados foram comentadas na Seção 5.1.

5.3.1 Estudo de Caso com o Projeto Genoma Pb

No Projeto Genoma Pb a base de dados utilizada contém os resultados de anotação automática de 6.107 grupos, divididos em grupos de 2.662 *contigs* e 3.445 *singlets* (conforme apresentado no gráfico da Figura 5.5).

O gráfico da Figura 5.6 apresenta a distribuição das seqüências não anotadas e as anotadas manualmente que foram analisadas pelo *BioAgents*.

Para obter os resultados apresentados na Tabela 5.1, o *BioAgents* utilizou os agentes GR *BLAST* e *FASTA*, instanciando para a análise das bases de dados os agentes ANL conforme a seguinte divisão:

- agente GR *BLAST*: responsável pelo fluxo de controle dos agentes ANL designados para análise das bases *nr*, *xyva* (*COG*) e *GO*;
- agente GR *FASTA*: responsável pelo fluxo de controle dos agentes ANL designados para análise das bases SC e SP.

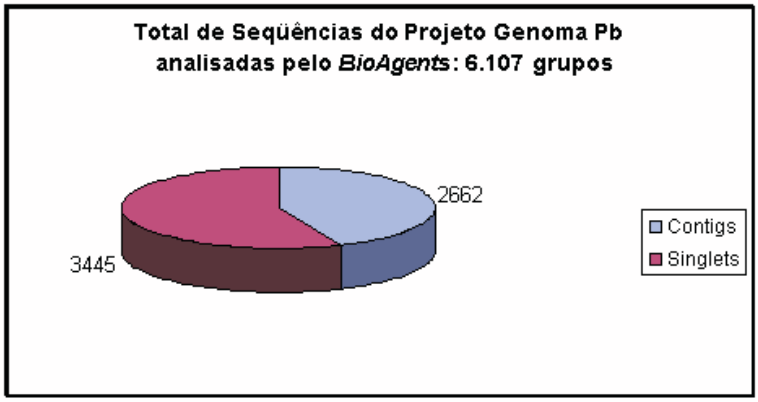


Figura 5.5: Distribuição do total de grupos do Projeto Genoma Pb.

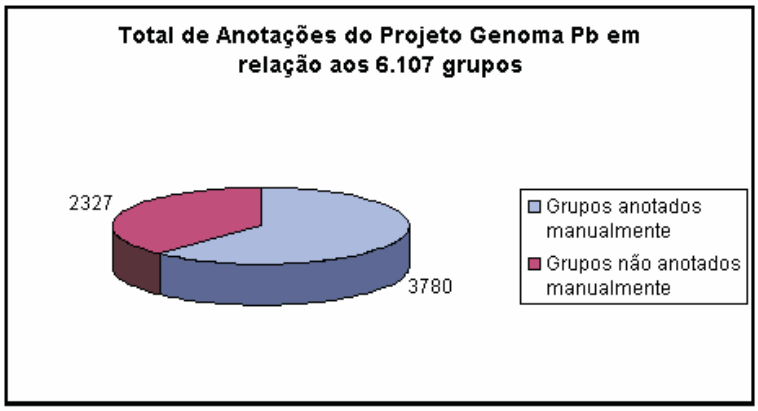


Figura 5.6: Distribuição do total de grupos do Projeto Genoma Pb não anotados e os anotados manualmente.

Desta forma, os agentes ANL são criados e destruídos (conforme apresentado na Seção 4.2) à medida que analisam cada um dos *contigs* e *singlets* que compõem os 6.107 grupos analisados.

A Tabela 5.1 reporta os resultados retornados pelo *BioAgents*. Observando esta tabela podemos verificar que 3.496 anotações foram sugeridas pelo *BioAgents*, sendo que 1.547 foram sugestões corretas quando comparadas com as anotações manuais do Projeto Genoma Pb. Em relação as 1.949 sugestões não corretas quando comparadas com as anotações manuais deste projeto, 336 foram sugestões a seqüências não anotadas e 1.613 foram sugestões diferentes das anotadas pelos biólogos. O gráfico da Figura 5.7 apresenta a composição das sugestões fornecidas pelo *BioAgents*.

Tabela 5.1: Resultados dos testes com o Projeto Genoma Pb (adaptada de [107]).

Parâmetro	**Valor**
Quantidade de genes	6.107
Quantidade de genes anotados manualmente	3.780
Quantidade de anotações sugeridas pelo *BioAgentes*	3.496
Quantidade de anotações acertadas pelo *BioAgentes*/ Quantidade de anotações sugeridas(% de acerto)	1.547/3.496 44.25%
Quantidade de anotações sugeridas para seqüências não anotadas	336
Tempo de execução do *BioAgents* (hh:mm)	01:30

Conforme comentado em [107], o tempo de execução longo (01h:30m) apresentado na Tabela 5.1 foi motivado pelo fato da base de dados referente ao Projeto Genoma Pb ser formada apenas por arquivos texto com tamanho aproximado de 1.5GB.

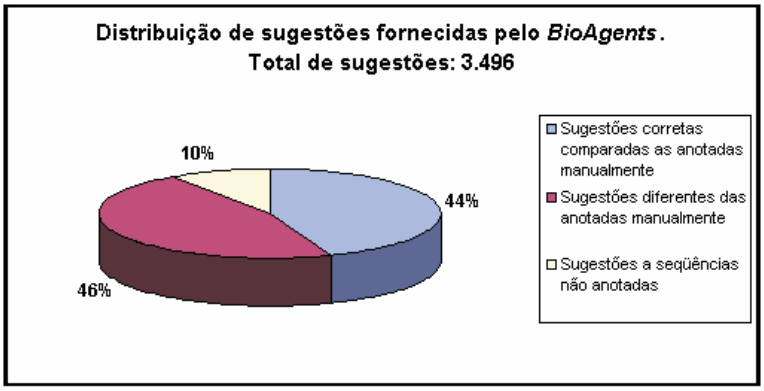


Figura 5.7: Distribuição das sugestões fornecidas pelo *BioAgents* ao Projeto Genoma Pb.

5.3.2 Estudo de Caso com o Projeto Genoma Guaraná

Em relação ao Projeto Genoma Guaraná a base de dados utilizada contém os resultados de anotação automática de 8.597 grupos, divididos em grupos de 2.628 *contigs* e 5.969 *singlets* (conforme apresentado no gráfico da Figura 5.8).

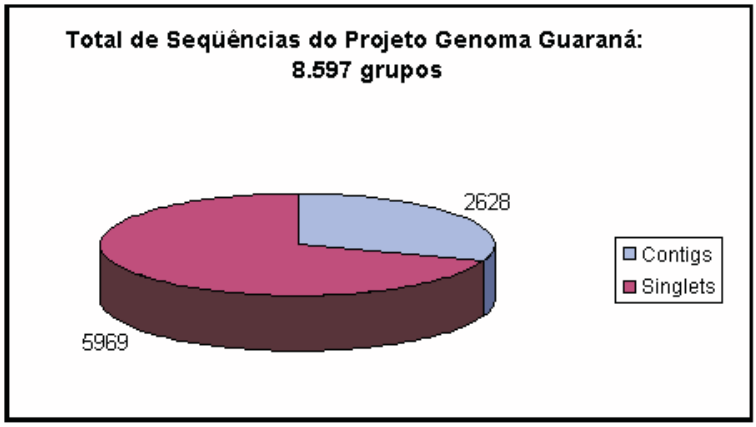


Figura 5.8: Distribuição do total de grupos do Projeto Genoma Guaraná.

Assim como no primeiro estudo de caso, o gráfico da Figura 5.9 apresenta a distribuição dos grupos não anotados e os anotados manualmente que foram analisadas pelo *BioAgents*.

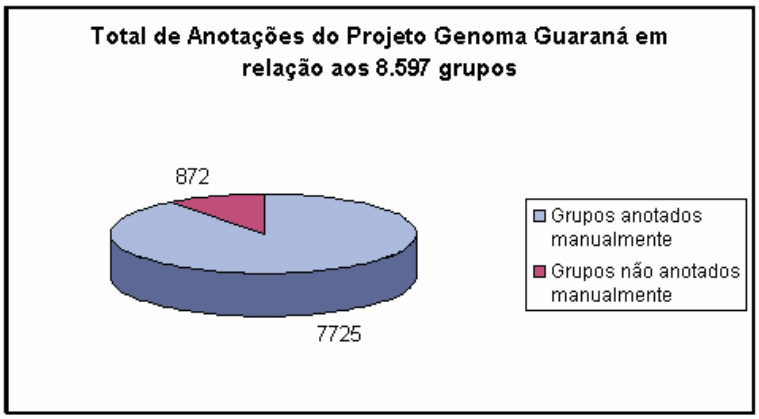


Figura 5.9: Distribuição do total de grupos do Projeto Genoma Guaraná não anotados e os anotados manualmente.

No estudo de caso do Projeto Genoma Pb observamos a utilização dos agentes GR *BLAST* e *FASTA* para obter os resultados. No que se refere ao Projeto Genoma Guaraná, o *BioAgents* utilizou apenas o agente GR *BLAST*, devido a anotação automática deste projeto não ter feito uso do programa *FASTA*. O agente GR *BLAST* é responsável pelo fluxo de controle dos agentes *ANL* designados para análise das bases *nr*, *KOG*, *GO* e *SwissProt*. A execução do agente GR e dos agentes ANL é análogo ao apresentado no primeiro estudo de caso.

O *BioAgents* sugeriu 6.478 anotações neste estudo de caso, das quais 2.938 constituiram sugestões corretas em comparação com as anotações manuais do Projeto Genoma Guaraná conforme Tabela 5.2. A distribuição dos resultados é apresentada igualmente ao primeiro estudo de caso, ou seja, no que refere-se as 3.540 sugestões não corretas quando comparadas com as anotações manuais, 231 foram sugestões a seqüências não anotadas e 3.309 foram sugestões diferentes das

anotadas pelos biólogos. O gráfico da Figura 5.10 apresenta a composição das sugestões fornecidas pelo *BioAgents*.

Tabela 5.2: Resultados dos testes com o Projeto Genoma Guaraná.

Parâmetro	**Valor**
Quantidade de genes	8.598
Quantidade de genes anotados manualmente	7.725
Quantidade de anotações sugeridas pelo *BioAgentes*	6.478
Quantidade de anotações acertadas pelo *BioAgentes*/ Quantidade de anotações sugeridas(% de acerto)	2.938/6.478 45.35%
Quantidade de anotações sugeridas para seqüências não anotadas	231
Tempo de execução do *BioAgents* (hh:mm)	04:05

Nota-se um maior tempo de execução (04h:05m), se comparado ao primeiro experimento, devido a fatores como o formato e armazenamento dos arquivos, pois os dados do Projeto Genoma Guaraná estão em formato *html*, armazenados em um banco *PostgreSQL*, para o qual a tabela possui um tamanho de aproximadamente 1.7GB. Desta forma, como utilizamos *parsers* (conforme descrito na Seção 4.2.2) para extrair deste grande volume de dados os campos necessários à analise do *BioAgents*, o custo de execução torna-se maior.

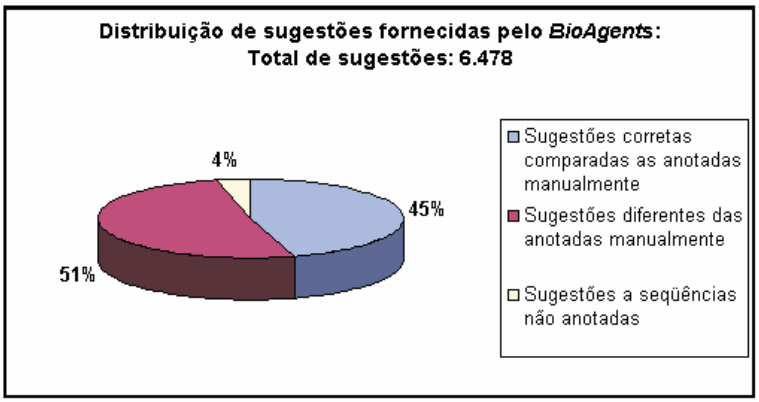


Figura 5.10: Distribuição das sugestões fornecidas pelo *BioAgents* ao Projeto Genoma Guaraná.

5.3.3 Análise Geral e Outros Resultados

Em relação ao tempo de execução do sistema podemos observar um tempo de processamento bem distinto nos dois estudos de caso realizados. Isto deve-se principalmente ao conteúdo das bases utilizadas no Projeto Genoma Pb serem constituídos em sua totalidade de arquivos texto. No caso dos dados do Projeto Genoma Guaraná, estes estão armazenados em um banco relacional, mas em formato *html*. Desta maneira, percebe-se que a utilização de *parsers* resultou num dispêndio do tempo de processamento, que poderia ser minimizado caso os arquivos de saída estivessem em um formato padrão, como por exemplo *xml* ou todos

os dados fossem atributos de uma tabela relacional. Entretanto, é interessante observarmos este fato a partir de um outro ponto de vista, ou seja, em relação ao tempo utilizado para executar uma anotação manual. Este tempo varia de acordo com as diversas características relacionadas ao projeto, bem como o nível de experiência dos anotadores. Por exemplo, no caso do Projeto Genoma Pb os quarenta biólogos responsáveis pela anotação manual levaram aproximadamente quarenta e cinco dias para executar esta tarefa. Sendo assim, o *BioAgents* pode auxiliar adequadamente nesta questão.

Em relação ao quantitativo de acertos computados pelo *BioAgents*, tanto quanto sabemos, não há trabalhos que indiquem uma porcentagem de acerto estabelecida para a anotação manual. Isto deve-se principalmente ao fato desta tarefa não apenas necessitar de conhecimento explícito, requerendo em diversos momentos a habilidade tácita (o conhecimento implícito dependente de representação e raciocínio complexo) dos anotadores. Conforme avaliação dos biólogos integrantes do Laboratório de Bioinformática da UnB, os resultados alcançados pelo *BioAgents* são bons e podem ainda ser melhores à medida que for especializada a *expertise* dos agentes. Observando este contexto, julgamos que o *BioAgents* pode ser um mecanismo de auxílio aos biólogos para a execução da tarefa da anotação manual. Três publicações foram obtidas a partir de 2005 e são apresentadas no Apêndice B.

5.4 Trabalhos Correlatos

Técnicas de IA têm sido abordadas por diversos trabalhos na área de Bioinformática, por meio do uso de abordagens distintas como a de SMA, arquiteturas *blackboard*, Mineração de Dados - observando o volume de dados crescentes nas bases de anotação - bem como aplicação de algoritmos de aprendizagem de máquina. Essas abordagens têm sido aplicadas em diferentes processos envolvidos no *pipeline* de execução, incluindo desde a comparação e análise de genomas até a inferência das funções dos genes dos organismos. Nessa seção apresentaremos trabalhos relacionados ao processo de anotação.

O *Eletronic Annotation* (*EAnnot*) é uma ferramenta originalmente desenvolvida para automatizar a tarefa de anotação manual do genoma humano, baseado em modelos de predição de genes [75]. O *software* combina ferramentas para extrair e analisar grandes volumes de dados em bases públicas, gerando anotações automáticas e predições de genes. *EAnnot* constrói modelos de genes (*genes models*) baseados em *mRNA*, *EST* e alinhamentos de proteínas para seqüências de genomas, suporta evidências para genes correspondentes, identifica pseudogenes, entre outras características.

O modelo do *EAnnot* que fornece as anotações manuais é baseado na utilização de *softwares* de predição gênica, bem como em um algoritmo implementado pelos autores, o qual considera para as predições diversas características presentes em um genoma. No entanto, as características deste algoritmo não são apresentadas. Segundo [75], o *EAnnot* pode ser aplicado para auxiliar a anotação manual de genomas de eucariotos. Para verificar a acurácia do *software*, o conjunto de genes preditos pelo *EAnnot* foi comparado ao conjunto de genes do cromossomo

6 do genoma humano, anotados manulamente por membros do *WTSI*. O *EAnnot* possui uma *interface Web* para verificação de resultados e é distribuído sob licença *GPL*.

Comparando ao escopo do protótipo apresentado na Seção 4.2, ao contrário do *EAnnot*, o *BioAgents* não utiliza algoritmos e/ou sistemas de predição para sugerir anotações manuais. O *BioAgents* sugere as anotações a partir da análise dos arquivos de saída do processo de anotação automática com os resultados dos *softwares* de comparação *BLAST* (programa *BLASTX*) e *FASTA*, para os quais os agentes possuem uma *expertise* representada por regras de produção. O *BioAgents* atualmente opera com a anotação de *EST*, enquanto o *EAnnot* avalia estratégias mais complexas, como os modelos de genes. Entretanto, o *BioAgents* é modular o suficiente para que seus agentes sejam especializados de acordo com as características de anotação desejáveis pelos biólogos em diferentes projetos de seqüenciamento de genomas, enquanto o *EAnnot* depende de algoritmos de predição específicos aplicados em determinados genomas.

O sistema *GeneQuiz* foi desenvolvido para análise de seqüências abrangendo desde a seqüência de proteína até sua função bioquímica, utilizando uma variedade de aplicativos e de fontes de dados de proteínas e *DNA* [45]. Fazendo um paralelo com os sistemas do tipo *pipeline* apresentados na Seção 2.4.3, podemos considerar o *GeneQuiz* como um *pipeline* exclusivo à anotação de seqüências, isto porque ele não aborda as questões relacionadas a submissão e montagem de seqüências. O sistema *GeneQuiz* é composto de quatro módulos, a saber:

- *GQupdate*: gerencia e atualiza a base do *GeneQuiz* a partir das seqüências disponíveis em bases públicas.
- *GQsearch*: analisa, por meio de diversos métodos e programas disponibilizados na literatura, as seqüências submetidas ao sistema. Os resultados reportados pelos programas são armazenados em diretórios próprios do sistema.
- *GQreason*: é utilizado para atribuir funções e outras características, por exemplo, número de proteínas homólogas às seqüências. Considerado o módulo principal do sistema, *GQreason* utiliza regras baseadas no conhecimento dos biólogos para avaliar os dados do sistema.
- *GQbrowse*: apresenta os resultados aos usuários, bem como provê acesso a bases de dados externas a partir de uma *interface Web* própria.

O foco do sistema *GeneQuiz* é gerenciar as diversas atividades que podem ser executadas em uma anotação de seqüências, utilizando para isso diversos métodos, *softwares* e bases de dados públicas. Esta abordagem é diferente da utilizada no protótipo *BioAgents*, pois este foi projetado para atuar na tarefa de anotação manual, ou seja, deixando a execução e o gerenciamento dos diversos métodos, *softwares* e bases de dados sob responsabilidade do sistema de *pipeline* utilizado pelo projeto de seqüenciamento relacionado. Entendemos que este fato seja uma vantagem do *BioAgents*, pois os agentes e as fontes de conhecimento podem ser especializadas a fim de tornar o sistema adequado a atender aos objetivos específicos de um determinado projeto de anotação manual. Por outro lado,

o uso de regras baseadas no conhecimento dos biólogos é um ponto comum entre o *BioAgents* e o *GeneQuiz*, mas não encontramos em [45] e [37] a estrutura das regras utilizadas pelo *GeneQuiz*.

Observando a utilização do paradigma de agentes, o trabalho de [73] apresenta o *software* denominado *BioMAS*. Este *software* é um SMA originalmente desenvolvido para a anotação automática do vírus da herpes. O *framework* utilizado para desenvolvimento foi o *DECAF* [38], desenvolvido pelos próprios autores do *BioMAS*. O foco deste sistema está na extração da informação contida nas bases de dados do vírus da herpes e no processo de anotação automática, que envolve a integração e análise destas informações. O *BioMAS* está dividido em três módulos, sendo que o primeiro corresponde a anotação básica de seqüências, o segundo objetiva a anotação funcional e o terceiro está relacionado ao processamento de *EST*. Existem três categorias principais de agentes que compõem a arquitetura do *BioMAS*, a saber:

- Agentes de Extração de Informação (*Information Extraction Agents*): são os agentes responsáveis por proverem acesso as bases de dados externas.

- Agentes de Tarefa (*Task Agents*): são os agentes responsáveis por recuperar, analisar as informações das bases de dados e direcionar aos biólogos quais informações deverão ser anotadas.

- Agentes de *Interface* (*Interface Agents*): são os agentes responsáveis por interagirem com o usuário final, bem como com outros agentes do sistema.

O *BioMAS* é uma abordagem de SMA bastante diversificada no que se refere ao apoio à anotação, tanto automática quanto manual. Uma outra forte característica deste sistema sobre o protótipo apresentado nesta dissertação é sua execução em um ambiente distribuído, o que reduziu bastante o tempo de execução do sistema se comparado ao *BioAgents*. No entanto, sugerimos a futura execução distribuída do *BioAgents*. Assim como a comparação do *BioAgents* com os sistemas *GeneQuiz* e *EAnnot*, o *BioMAS*, apesar de dar cobertura ao tema, não realiza a tarefa de anotação manual. Uma outra colocação que deve ser feita, é a forma na qual o *BioMAS* trabalha. Este sistema, assim como o *EAnnot*, trabalha com predição de genes, ou seja, o sistema busca apoiar a anotação baseado em modelos de genes, o que não é o caso do *BioAgents*, que nesta dissertação propôs sugestões a anotações manuais de *EST*, que é considerado um universo menor se comparado aos modelos de genes. Estes fatos adicionados a estratégia de sugestões adotada pelo *BioAgents* são dois pontos fortes de diferença entre nossa abordagem e o *BioMAS*.

O *Agent-based environmenT for a aUtomatiC annotation of Genomes* (*ATUCG*) [54] é um sistema baseado em agentes, tendo como objetivo apoiar o processo de anotação automática de genomas utilizando estratégias de aprendizagem de máquina. A arquitetura do *ATUCG* é divida em três camadas. A primeira camada é responsável por auxiliar os usuários a identificarem *ORF* a partir da entrada de uma seqüência de *DNA*, utilizando-se agentes que são responsáveis pela execução de *softwares* de predição gênica, tais como o *Glimmer* [39] e o *Genemark* [40]. Na segunda camada agentes especializados na extração de dados de bases

públicas (por exemplo, *TrEMBL* e *Swiss-Prot*) interagem com agentes de treinamento que utilizam algoritmos de apredizagem de máquina, como o *C4.5* [125], e identificam regras que são posteriormente utilizadas para a anotação de campos da base de dados *Swiss-Prot*. A terceira camada consiste da apresentação das anotações aos usuários, a fim de que estes possam validá-las. Após a validação, as anotações são armazenadas em dois bancos de dados administrados pelo sistema, a saber: banco de anotações aceitas e banco de anotações rejeitadas.

Integrando a proposta de [54], o trabalho de [118] desenvolve um módulo - integrado a arquitetura do *ATUCG* - para o problema da re-anotação. Este módulo é baseado em agentes que avaliam uma série de *ORF* (anteriormente anotadas e mantidas pela base de anotações aceitas do *ATUCG*) com bases de dados públicas, como o *GenBank* [57]. Desta forma, o módulo de re-anotação compreende agentes especializados que executam *softwares* como o *BLAST*, *Glimmer* e o *Genemark* a fim de identificar novas informações que possam existir e agregar valor às anotações anteriormente realizadas. Caso sejam identificados novas anotações, o sistema diponibiliza-as aos especialistas a fim de que estes as confirmem, para que a base de anotações seja atualizada.

No trabalho de [54], as regras utilizadas para as anotações são derivadas de algoritmos de aprendizagem de máquina, os quais utilizam dados de anotações contidas nas bases *Swiss-Prot* e *TrEMBL* a fim de construir as bases de treinamento para estes algoritmos. No protótipo *BioAgents* as regras que servem de *expertise* aos agentes procuram representar o conhecimento biológico que o anotador expressa na execução da anotação manual. Uma outra diferença, igualmente reportada aos projetos descritos anteriormente, refere-se ao fato de que o *BioAgents* utiliza informações recuperadas diretamente da anotação automática, ou seja, todo o *overhead* da execução de algoritmos como *BLAST* e *FASTA* é deixado sob responsabilidade do sistema do tipo *pipeline* utilizado para executar a anotação automática. Este fato faz com que o *BioAgents* seja especificamente focado na execução da tarefa de anotação manual.

Em relação ao trabalho de [118], esta abordagem é conceitualmente diferente do *BioAgents*, por este tratar da anotação manual e aquele do problema da re-anotação. Entretanto, a anotação manual e a re-anotação não são duas atividades disjuntas, isto é, a medida que auxiliarmos os biólogos a melhor executar a anotação manual, a re-anotação estará focada cada vez mais na busca somente por novas informações que agregam valor às anotações anteriormente efetuadas, de forma a não se preocupar em desenvolver metodologias para identificar erros que podem ser tratados diretamente na anotação manual. Estes erros são minimizados na anotação manual principalmente pela acurácia dos resultados reportados pelos *softwares* utilizados na tarefa de anotação automática, sendo que esta acurácia é determinada principalmente pelas baixas ocorrências de falsos negativos (resultados não fornecidos mas que constituem informações biologicamente verdadeiras) e falsos positivos (resultados fornecidos mas não correspondentes a informações biologicamente verdadeiras) nestes resultados.

A Tabela 5.3 mostra a comparação entre os trabalhos discutidos e o protótipo *BioAgents* apresentado nesta dissertação. As características relacionadas seguem a seguinte classificação:

- S: a característica está implementada;
- S*: a característica está proposta mas não implementada;
- N: não possui a característica.

Tabela 5.3: Comparação entre os trabalhos correlatos discutidos.

	EAnnot	***GeneQuiz***	***BioMAS***	***ATUCG***	***BioAgents***
Modelos de Predição Gênica	S	S	S	S	N
Técnicas de Mineração de Dados	N	N	N	S	S*
Algoritmos de Aprendizagem de Máquina	N	N	N	S	S*
Representação do Conhecimento	N	S	S	S	S
Apoio ao Processo de Anotação	S	S	S	S	S
Foco específico na Anotação Manual	N	N	N	N	S
Re-anotação	N	N	N	S	N

Na Tabela 5.3, relacionamos sete principais características identificadas entre os trabalhos correlatos e o trabalho apresentado nessa dissertação, a saber: (i) Modelos de Predição Gênica, (ii) Técnicas de Mineração de Dados (pré-processamento dos dados), (iii) Algoritmos de Aprendizado de Máquina (por exemplo, o algoritmo *C4.5*), (iv) Representação do Conhecimento (por exemplo, as regras de produção utilizadas na representação de conhecimento biológico), (v) Apoio ao Processo de Anotação (automática e/ou manual), (vi) Foco Específico na Anotação Manual (sugerir anotações com base em resultados obtidos pelos biólogos) e (vii) Re-Anotação (apoio as possíveis atualizações ocorridas na anotação de genomas após as anotações automática e manual).

Neste capítulo foram apresentados dois estudos de caso envolvendo os projetos genoma Pb e Guaraná, com a finalidade de validar a proposta da arquitetura e sua respectiva implementação. Foram discutidos os resultados alcançados nos estudos de caso. Também foram apresentados e discutidos alguns trabalhos correlatos mais relacionados a proposta deste trabalho.

Capítulo 6

Conclusões e Trabalhos Futuros

Nessa dissertação discutimos a aplicabilidade da abordagem de SMA para a tarefa de anotação manual em projetos de seqüenciamento de genomas. A partir de um estudo direcionado à bionformática e a agentes inteligentes, foi apresentado a proposta de um SMA especificamente focado na anotação manual, utilizando técnicas de mineração de dados a fim de definir padrões encontrados na anotação automática que auxiliem na anotação manual, bem como a construção de ontologias específicas que possam relacionar conceitos existentes neste domínio [105].

Com o objetivo de explorar o potencial da proposta, apresentamos o desenvolvimento de um protótipo denominado *BioAgents* [107, 106]. Este protótipo foi desenvolvido utilizando metodologia baseada em SMA [143, 152, 157, 129, 112], apresentando uma arquitetura multiagente implementada conforme uma abordagem *blackboard* [121, 70, 71]. Nesta arquitetura, agentes especializados em tarefas distintas interagem com bases de dados próprias e fontes de conhecimento específicas, baseadas em regras de produção, para fornecer sugestões de anotações manuais aos biólogos. O protótipo *BioAgents* está implementado na linguagem *Java* [27], utilizando o suporte do *framework JADE* [55, 56] para desenvolvimento de SMA.

Para validar o protótipo, realizamos dois estudos de caso. No primeiro utilizamos os dados do Projeto Genoma Pb e no segundo os dados do Projeto Genoma Guaraná. Com a *expertise* dos agentes sendo representada por poucas regras de produção, alcançamos, respectivamente aos projetos de estudo de caso descritos, indíces de acerto de anotação manual correspondentes a 44.25% e 45.35%. Estes indíces foram obtidos comparando diretamente o número de sugestões corretas fornecidas pelo *BioAgents* com as anotações manuais anteriormente executadas nos projetos genoma Pb e Guaraná. O *BioAgents* também forneceu para o Projeto Genoma Pb 336 sugestões, e para o Projeto Genoma Guaraná 231 sugestões, ambas a grupos não anotados. Estas sugestões foram consideradas corretas após análise amostral dos biólogos.

Há diversas abordagens que por meio da anotação automática apoiam a tarefa de anotação manual, utilizando e executando *softwares* disponíveis para comparação de seqüências, predição e identificação de modelos de genes, entre outros. Entre estas abordagens, inclusive a de SMA, existem vários trabalhos que agregam aos métodos da anotação automática técnicas como o uso de algoritmos de aprendizagem de máquina, buscando, por exemplo, melhorar a *expertise* dos agen-

tes através de regras de decisão [54]. Alguns destes trabalhos [75, 45, 73, 54, 118] foram discutidos nesta dissertação e comparações deles com o *BioAgents* foram realizadas.

Não é nosso objetivo direcionar qual a melhor abordagem a ser utilizada, nem como sugerir qual é a melhor ferramenta. Julgamos que o *BioAgents* contribui por focar especificamente na tarefa de anotação manual. Mas a escolha depende de características e objetivos específicos de cada projeto de seqüenciamento, e cada uma das abordagens pode ser mais útil em um determinado caso.

O protótipo *BioAgents* possui limitações, pois a finalidade deste trabalho foi de validar a proposta de SMA aplicada à tarefa de anotação manual. Podemos citar como limitações básicas a *interface Web* e a execução distribuída, porém restrita a uma única máquina, dos agentes do SMA. Assim, há boas pespectivas de trabalhos futuros a serem executados, de forma a dar continuidade ao trabalho apresentado nesta dissertação, a saber:

- Aprimorar a execução distribuída dos agentes, buscando mecanismos que possam direcionar aos agentes ANL os recursos computacionais disponíveis a fim de que as tarefas de análise, as quais usam *parsers*, sejam realizadas de forma distribuída, através do armazanamento dos dados em formato *xml*, reduzindo sensivelmente o tempo de execução total e viabilizando o uso do *BioAgents* em diversas instituições;

- Desenvolver o mecanismo de resolução de conflitos do *BioAgents*, a fim de gerenciar os reais conflitos que possam existir entre as sugestões reportadas pelos agentes GR, principalmente quando da inclusão de novas bases de dados (de acordo com a interpretação apresentada na Seção 5.1);

- Aumentar a *expertise* representada nas regras utilizadas pelos agentes, incluindo, por exemplo, uma regra que use a informação da base de dados que está originando um determinado alinhamento (*hit*). Para os biólogos, a confiabilidade da base que originou um determinado *hit* é muito importante para melhor anotar uma seqüência;

- Especializar novas FC, desenvolvendo regras que trabalhem com informações fornecidas para detecção de *ncRNA* (*INFERNAL*), identificação de *tRNA* (*tRNAscan-SE*), identificação de homologias em famílias de proteínas (*HMMER/Pfam*), bem como acrescentar os conceitos relacionados a estas informações nos vocabulários que compõem o componente BB desenvolvido;

- Desenvolver o módulo colaborativo descrito na proposta apresentada neste trabalho, estendendo o vocabulário utilizado à uma ontologia para as bases de dados que venham a integrar o *BioAgents*, bem como definir e implementar processos e algoritmos de mineração de dados para descoberta de padrões, implícitos nos resultados da anotação automática, que ampliem a inteligência dos agentes na tarefa de anotação manual de forma a completar a proposta da arquitetura de SMA. O trabalho de [54] traz uma abordagem interessante ao auxílio na execução deste trabalho;

- Utilizar o protótipo em projetos de seqüenciamento de genomas que não tenham executado a tarefa de anotação manual, como o Projeto Genoma Anaplasma (comentado nas Seções 2.2.1 e 2.4.2) para acelerar a execução da anotação;

- Integrar o protótipo *BioAgents* com ao *framework* Timina [69]. Esta atividade encontra-se em execução, estando atualmente definida a *interface* de comunicação entre estas duas ferramentas;

- Desenvolver um módulo de aprendizagem a partir das recomendações de anotação manual feitas pelo *BioAgents*.

Enfim, são muitas as possibilidades de estender os resultados alcançados neste trabalho de mestrado, e isto certamente vem comprovar o uso deste enfoque em Bioinformática. Vale ressaltar que a proposta tal como se encontra tem a característica principal de redução do esforço humano, durante o processo de anotação manual, pois grande parte das anotações diretamente provenientes da anotação automática podem ser feitas de forma correta. Também é importante ressaltar que com um aprofudamento no conhecimento dos biólogos expressos pelas regras de produção, o sistema *BioAgents* poderá recomendar anotações manuais realizadas somente pelos biólogos expertos reduzindo muito o esforço humano.

Apêndice A

Código das Regras de Produção

Incluimos nesse apêndice o código fonte das regras de produção implementadas em *JESS*, utilizadas para representar a *expertise* do protótipo *BioAgents*. As regras implementadas e utilizadas pelo agente GR e pelos agentes ANLs *FASTA* são idênticas as utilizadas pelos agentes *BLAST*, isso porque a definição das regras junto aos biólogos do Laboratório de Bioinformática da UnB foi igualmente aplicada à análise dos dados resultantes dos *softwares BLAST* e *FASTA*.

```
(reset)

;;; Module MAIN ;;;

(deftemplate BlastHit
  (declare (slot-specific TRUE))
  (slot HitAccession (type STRING))
  (slot HitID (type STRING))
  (slot HitLength (type INTEGER))
  (slot HitNumber (type STRING))
  (slot HitDefinition (type STRING))
  (slot HitEvalue (type FLOAT))
  (slot HitScore (type FLOAT))
)


(deftemplate EvalueAnalysis
    (slot HitID  (type STRING))
    (slot Evalue (type FLOAT))
    (slot Score  (type INTEGER))
)

(defglobal ?*maxEvalue* = 1.0E-5);

(defquery FindEvalueAnalysis

    (declare (variables ))
```

```
    (EvalueAnalysis(HitID ?hitId))
    )

(deffunction start ()

    (set-current-module MAIN)
    (focus Evalue)
    (run)
)

;;; Module Evalue ;;;

(defmodule Evalue)

(defrule Exists_Evalue_Below_Limit

    (exists (BlastHit
                (HitEvalue ?evalue&:(<= ?evalue ?*maxEvalue*))
    ))
    =>
    (focus GoodEvalueAnalysis)
    (run)
)

(defrule All_Evalue_Above_Limit

    (not (BlastHit
                (HitEvalue ?evalue&:(<= ?evalue ?*maxEvalue*))))
    =>

)

;;;  Module GoodEvalueAnalysis ;;;

(defmodule GoodEvalueAnalysis)

(defrule Assert_Evalues_Facts
    "comment"
    (logical (BlastHit
                (HitID ?id)(HitEvalue ?evalue&:(<= ?evalue ?*maxEvalue*))
                           (HitScore ?score)))
    =>
    (assert (EvalueAnalysis(Evalue ?evalue)(Score ?score)(HitID ?id)))
)

(defrule Best_Evalue
    "comment"
```

```
    (EvalueAnalysis(Evalue ?evalue1)(Score ?score1))
    ?fact <- (EvalueAnalysis(Evalue ?evalue2&:(>= ?evalue2 ?evalue1))
                             (Score ?score2&:(< ?score2 ?score1)))
    =>
    (retract ?fact)
)
```

Apêndice B

Publicações Obtidas

Nesse apêndice são apresentadas as publicações obtidas com o trabalho apresentado nesta dissertação.

B.1 Primeira publicação

- A Multiagent System to Help Manual Annotation on Genome Sequencing Projects.
  - Autores: Richardson Silva Lima, Célia Ghedini Ralha, Maria Emilia Machado Telles Walter e Marcelo de Macedo Brígido.
  - Evento: International Workshop on Genomic Databases (IWGD).
  - Local e Data de Realização: Rio de Janeiro, RJ, Brasil. 10 a 11 de novembro de 2005.
  - Tipo da publicação: Resumo expandido.
  - Idioma: Inglês.
  - Breve Resumo: Este trabalho apresentou a proposta, conforme descrita (com adequações) nesta dissertação, hipótese relacionada e o direcionamento para o desenvolvimento de um protótipo de validação utilizando as bases de dados do Projeto Genoma Pb.

Íntegra do artigo:

Nowadays, there is an enormous amount of biological data information available on the Web. Biologists use this information to infer functions of genes discovered on genome sequencing projects. This task constitutes the annotation phase for which automatic tools can greatly help specialists to improve the accuracy of the annotation. The motivation of this work is to mimic biologists interaction in order to improve manual annotation on genome sequencing projects using artificial intelligence techniques.

The hypothesis of this project is that biological data sources developed by autonomous research groups differ on their ontological commitments. These include assumptions concerning the existing objects, the properties or attributes of the objects, relationships among objects, the possible values of attributes, and their intended meaning, as well as the granularity or level of abstraction at which objects and their properties are described. Our proposal focus the fact that there is no single privileged ontology and mining algorithms that can serve all agents with the same reasoning mechanism.

This article presents a collaborative system with a multiagent approach, based on integrated, heterogeneous and autonomous genomic data sources available on the Web. The system combines different agents with specific ontologies and mining algorithms, which interact with each other and the environment, cooperating in order to automatically suggest annotation to be validated by biologists. The proposed architecture consists of a three-layer platform with the physical layer, the collaborative agent environment layer, and the user-interface layer. The physical layer allows the system to communicate with the different information sources based on federated database architectures. The collaborative agent environment layer automatically transforms queries from users into execution plans to the agents, defining what information to extract and how to combine the results to suggest annotation, including the conflict resolution among the different agents. The user-interface layer enables users to interact with the system, post queries and receive answers. We defined agents to acquire rules for annotation from databases used by BLAST, PFAM, PSORT and INFERNAL. In order to validate the annotation suggested by the multiagent system, comparisons against already manually annotated genes of the Paracoccidioides brasiliensis fungus - the organism sequenced in the midwest region of Brazil will be made (`https://www.biomol.unb.br/Pb/`).

References

1. A Bateman et al. **The Pfam Protein Families Database**. *Nucleic Acids Research*, 30(1), p. 276-280, 2002.

2. A. L. C. Bazzan et al. **ATUCG-An Agent-Based Environment for Automatic Annotation of Genomes**. *International Journal of Cooperative Information Systems*, 12, p. 241-273, 2003.

3. A. Levy. **Logic-based Techniques in Data Integration**. *Logic Based Artificial Intelligence, Edited by Jack Minker*. Kluwer Publishers, 2000.

4. B. Eckman. **A Practitioner´s Guide to Data Management and Data Integration in Bioinformatics**. *Bioinformatics*, p. 3-74, 2003.

5. C. Lagoze and J. Hunter. **The ABC Ontology and Model**. *Journal of Digital Information*, 2 (2), 2001.

6. D. Caragea, J. Pathak, J. Bao, A. Silvescu, C. Andorf, D. Dobbs and V. Honavar. **Information Integration and Knowledge Acquisition from Semantically Heterogeneous Biological Data Sources**. In: *Proceedings of the 2nd International Workshop on Data Integration in the Life Sciences (DILS 2005)*. Edited by B. Ludäscher and L. Raschid: LNBI 3615, Springer-Verlag Berlin Heidelberg, p 175-190, 2005.

7. K. Nakai and P. Horton. **PSORT: a program for detecting the sorting signals of proteins and predicting their subcellular localization**, *Trends Biochem. Sci*, 24(1), p. 34-35, 1999.

8. L. V. Nascimento and A. L. C. Bazzan. **An Agent-Based System for Re-annotation of Genomes**. In: *Proceedings of the III Brazilian Workshop on Bioinformatics (WOB)*, Brasilia, 2004.

9. M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley Sons Ltd., 2002.

10. N. L. Harris. **Genotator: A workbench for sequence annotation**. *Genome Research*, 7(7), p. 754-762, 1997.

11. P. Gouret et al. **FIGENIX: Intelligent automation of genomic annotation: expertise integration in a new software platform**. *BMC Bioinformatics*, 6(198), 2005.

12. R. Karchin et al. **LS-SNP: Large-scale annotation of coding nonsynonymous SNPs based on multiple information sources**. *Bioinformatics*, 21(12), p. 2814-20, 2005.

13. R. Shaker, P. Mork, J. S. Brockenbrough, L. Donelson and P. Tarczy-Hornoch. **The Biomediator System as a Tool for Integrating Biologic Databases on the Web**. In *Proceedings of the Workshop on Information Integration on the Web (held in conjunction with VLDB 2004)*, Toronto, Canada; 2004.

14. S. F. Altschul et al. **Basic Local Alignment Search Tool**. *Journal of Molecular Biology*, 215, p. 403-410, 1990.

15. S. R. Eddy. **A memory efficient dynamic programming algorithm for optimal structural alignment of a sequence to an RNA secondary structure**. *BMC Bioinformatics*, 3(18), 2002.

B.2 Segunda publicação

- *BioAgents*: Um Sistema Multiagente para Anotação Manual em Projetos de Seqüenciamento de Genomas.

- Autores: Richardson Silva Lima, Célia Ghedini Ralha, Maria Emilia Machado Telles Walter, Hugo Wruck Schneider, Anderson Gray Frazzon Pereira e Marcelo de Macedo Brígido.
- Evento: Encontro Nacional de Inteligência Artificial (ENIA).
- Local e Data de Realização: Rio de Janeiro, RJ, Brasil. 03 a 06 de julho de 2007.
- Tipo da publicação: Artigo completo.
- Idioma: Português.
- Breve Resumo: Neste trabalho foi apresentado o protótipo *BioAgents* e sua arquitetura, apresentando resultados obtidos com o estudo de caso relacionado ao Projeto Genoma Pb. Alguns detalhes na arquitetura apresentada nesta dissertação, como a organização *blackboard* na *Camada Colaborativa*, bem como o estudo de caso com o Projeto Genoma Guaraná, não foram apresentados neste artigo.

Íntegra do artigo:

1. Introdução

Em 1953, Watson e Crick propuseram uma estrutura molecular para o *DNA* [Watson and Crick 1953]. Desde essa época, a comunidade científica vem dispendendo grandes esforços com o objetivo de compreender melhor a estrutura e o funcionamento da biologia molecular dos seres vivos. No início da década de 1990, foi iniciado o Projeto Genoma Humano, que visava mapear e seqüenciar, por completo, o genoma humano. Este projeto foi concluído em 2001 [Venter et al.2001, Lander et al. 2001], e apresentou o genoma humano com 3 bilhões de bases e aproximadamente 30.000 genes.

O Projeto Genoma Humano e inúmeros outros projetos de seqüenciamento de genomas surgidos em todo o mundo propiciaram grandes e rápidos avanços em técnicas da Biologia Molecular e Bioinformática [Liolios et al. 2006]. Assim, desde a década de 1990, podemos observar um crescimento exponencial no volume de dados gerados pelos diversos projetos de seqüenciamento de genomas. Em relação ao gerenciamento e análise destes dados, a área de Computação tem desenvolvido técnicas e *softwares* que apoiam o esforço dos biólogos no armazenamento e análise dos dados gerados nestes projetos. O sistema computacional que apoia estes projetos é denominado de *pipeline* ou *workflow* [Lemos 2004]. Um *pipeline* é dividido em três fases: submissão, montagem e anotação. A fase de submissão visa receber as seqüências geradas nos laboratórios de Biologia Molecular, transformando-as em cadeias de caracteres e armazenando-as em bancos de dados. A fase de montagem visa agrupar seqüências que potencialmente tenham vindo da mesma região do *DNA*. Cada grupo com mais de uma seqüência recebe o nome de *contig* e tem uma seqüência consenso que representa o grupo. Seqüências não agrupadas recebem o nome de *singlet*.

A fase de anotação tem o objetivo de inferir as funções biológicas das seqüências resultantes da montagem, utilizando funções conhecidas de seqüências similares disponibilizadas em bancos de dados biológicos. Esta fase é dividida em duas etapas: automática e manual. A anotação automática compara as seqüências geradas no projeto com seqüências de bancos de dados privados e/ou públicos (como o *GenBank* [Benson et al. 2006]). Métodos de comparação aproximada de seqüências[1] (como *BLAST* [Altschul et al. 1990] e *FASTA* [Pearson and Lipman 1988]) são utilizados para inferir funções das seqüências estudadas. Estas inferências são feitas comparando com seqüências semelhantes que tiveram suas funções previamente determinadas. Na anotação manual, os biólogos utilizam as informações da anotação automática, bem como seus conhecimentos, para determinar a função que deve ser associada à seqüência analisada.

Neste trabalho serão apresentados uma arquitetura e um protótipo de Sistemas Multiagente [Wooldridge 2002, Weiss 2000], denominado *BioAgents*, que visa auxiliar os biólogos na tarefa de anotação manual em projetos de seqüenciamento de genomas [Lima et al. 2005]. A escolha da abordagem Multiagente deve-se principalmente ao fato da aplicação apresentar características específicas adequadas ao uso desta tecnologia, a saber: (i) utiliza bancos de dados heterogêneos e descentralizados, (ii) constitui um ambiente dinâmico (por exemplo, novos tipos de dados e fontes de dados com constantes alterações), (iii) o processo de anotação pode ser realizado de forma independente por vários biólogos. A arquitetura apresentada foi implementada através de um protótipo que utiliza a plataforma de desenvolvimento de agentes *JADE* [Bellifemine et al. 2003], integrada ao motor de inferência *JESS* [Friedman-Hill 2003].

O protótipo foi utilizado em um estudo de caso que utiliza os dados do Projeto Genoma Funcional e Diferencial do fungo *Paracoccidioides brasiliensis* (Pb) [Felipe et al. 2005]. Esse projeto foi executado pela Rede Genoma Centro-Oeste, que integra instituições de ensino e pesquisa em Biologia Molecular do Distrito Federal, Goiás, Mato Grosso e Mato Grosso do Sul. As sugestões geradas automaticamente foram validadas através de comparações de resultados gerados pelo *BioAgents* com as anotações manuais previamente realizadas no Projeto Genoma Pb.

Este trabalho está dividido em cinco seções. Na seção 2 são mostrados alguns trabalhos correlatos. Na seção 3 é apresentada a arquitetura Multiagente e descrito o protótipo implementado. Na seção 4 o estudo de caso é apresentado, sendo feita uma breve discussão dos resultados. Na seção 5 concluímos e apresentamos trabalhos futuros.

2. Trabalhos Correlatos

Vários trabalhos na área de Bioinformática utilizam técnicas de Inteligência Artificial, através do uso de abordagens distintas como a de Sistemas Multiagente

[1]Dizemos que duas seqüências são similares quando partes delas são "aproximadamente iguais", isto é, quando as duas seqüências têm exatamente os mesmos caracteres, com poucas exceções de caracteres diferentes, ou inserções e remoções de caracteres de uma das seqüências em relação à outra.

(SMA), Mineração de Dados e/ou Aprendizagem de Máquina. Essas abordagens têm sido aplicadas em diferentes processos envolvidos no *pipeline* de execução, incluindo desde a comparação e análise de genomas até a inferência das funções dos genes dos organismos. Porém, não encontramos na literatura trabalhos que apliquem a abordagem de SMA para o processo de anotação manual. Apresentamos então trabalhos relacionados ao processo de anotação.

O sistema *BioMAS* utiliza a abordagem de SMA para anotação automática do vírus da herpes [Decker et al. 2001]. O foco do trabalho está na extração da informação contida nos bancos de dados públicos e no processo de anotação automática.

O *Eletronic Annotation-EAnnot* é uma ferramenta originalmente desenvolvida para a anotação manual do genoma humano [Ding et al. 2004]. O software combina ferramentas para extrair e analisar grandes volumes de dados em bancos públicos, gerando anotações automáticas e predições de genes de forma rápida. *EAnnot* usa informações contidas em *messenger RNA-mRNA*, *Expressed Sequence Tags-ESTs* e alinhamentos de proteínas, além de identificar pseudogenes, entre outras características.

O software *Ambiente para Anotação Automática e Comparação de Genomas-A3C* [Santos and Bazzan 2004] é baseado em uma aquitetura de SMA e tem como propósito a integração de tarefas relacionadas a anotação denominada pelos autores como nível 1 e a comparação de genomas considerada como nível 2. O nível 1 é composto por ferramentas para a anotação automática de proteínas; enquanto o nível 2 é composto por algoritmos para comparação de genomas que visam a extração de informações úteis aos resultados do nível 1. O objetivo do A3C é descobrir a relação entre diversos organismos, obtendo então informações específicas sobre um dado genoma através do conhecimento sobre outros genomas que já se encontram seqüenciados.

A ferramenta denominada *Agent-based environmenT for aUtomatiC annotation of Genomes-ATUCG* é baseada em uma aquitetura de agentes, tendo como objetivo reduzir o trabalho manual dos biólogos através da re-anotação [Nascimento and Bazzan 2005]. No processo de re-anotação as informações adquiridas das seqüências originalmente anotadas são revisadas e comparadas com novos modelos e dados para se obter características e informações sobre as seqüências e refazer a anotação manual, caso seja necessário.

3. A Arquitetura Multiagente e o Protótipo do *BioAgents*

Como dito anteriormente, o *BioAgents* visa auxiliar os biólogos no processo de anotação manual. O processo de anotação manual é executado pelos biólogos basicamente: analisando as saídas das ferramentas executadas durante o processo de anotação automática, e interpretando estes resultados, de acordo com seu conhecimento biológico, para inferir as funções e categorias funcionais das seqüências a serem anotadas. O *BioAgents* se propõe a simular esta tarefa dos biólogos.

A Figura B.1 representa a arquitetura SMA do *BioAgents*, que é composta por três camadas:

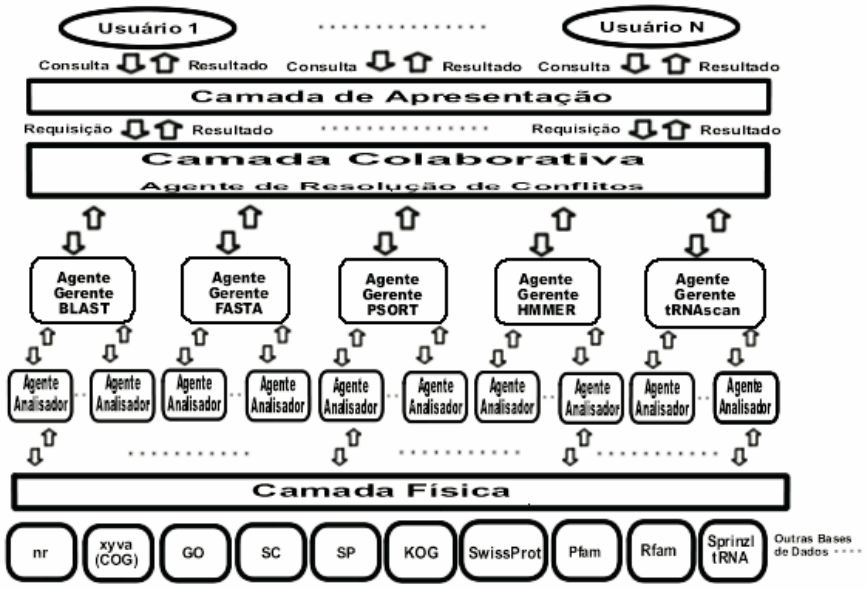


Figura B.1: A arquitetura em três camadas do sistema *BioAgents*.

- A *Camada de Apresentação* é responsável por receber as requisições submetidas ao sistema e retornar o resultado do processamento ao usuário. A requisição consiste na submissão de seqüências a serem analisadas. Na atual implementação, as ferramentas *BLAST* e *FASTA* e os bancos de dados utilizados, foram apenas informados para o sistema. Os arquivos de saída já processados constituíram a entrada para os *Agentes Analisadores* (ANL). Estes arquivos de saída contém os resultados das comparações efetuadas pelo *BLAST* e *FASTA*, tendo sido obtidos na etapa de anotação automática.

- A *Camada Colaborativa* é responsável pela consolidação dos resultados provenientes das análises feitas sobre os bancos de dados da *Camada Física* e por retorná-los à *Camada de Apresentação*. A *Camada de Colaboração* é composta pelo *Agente de Resolução de Conflitos* (RC), pelos *Agentes Gerentes* (GR) e pelos ANLs.

  - O agente RC tem o objetivo de submeter as requisições enviadas pela *Camada de Apresentação* aos agentes GR especializados. Após receber os resultados dos agentes GR, decide a sugestão mais apropriada para ser enviada à *Camada de Apresentação*. No estudo de caso realizado, foram utilizados os agentes GR e ANL *BLAST* e *FASTA*.

  - Os agentes GR recebem mensagens do agente RC com solicitações de acordo com sua especialidade. Um particular agente GR verifica quais são os bancos de dados e saídas dos programas que devem ter sido previamente executados na anotação automática. O agente GR aloca os agentes ANL para fazer a análise individual dessas saídas juntamente com os bancos de dados. O agente GR aguarda as sugestões de todos os agentes ANL, consolidando-as através do uso das regras

de produção previamente definidas. Como cada agente GR é especializado em um programa, ele pode avaliar e consolidar os resultados retornados pelos agentes ANL.

  - Cada agente ANL executa um arquivo de saída gerado por uma ferramenta específica. Quando é criado por solicitação de um agente GR, cada agente ANL utiliza um *parser* específico para extrair informações do arquivo de saída, gerando uma estrutura contendo dados específicos da ferramenta. O resultado desse processamento com a sugestão é retornada ao agente GR solicitante.

- A *Camada Física* é responsável pelos bancos de dados utilizadas pelo *BioAgents*. Em nosso estudo de caso foram utilizadas as seguintes fontes de dados: *nr-GenBank* (`http://www.ncbi.nlm.nih.gov/Genbank/`); *Gene Ontology* (GO) (`http://www.geneontology.org/`); *Clusters of Orthologous Groups of proteins* (COG) (`http://www.ncbi.nlm.nih.gov/COG/`) e os bancos de dados dos fungos *Saccharomyces cereviseae* (SC) e *Schizosaccharomyces pombee* (SP).

3.1. O Protótipo

Para implementar a arquitetura SMA proposta, utilizamos a linguagem *Java* (`http://java.sun.com`) no ambiente de desenvolvimento *Eclipse SDK*, versão 3.1.2 (`http://www.eclipse.org`). Como *framework* de desenvolvimento de agentes, utilizamos o *Java Agent DEvelopment Framework-JADE* versão 3.4.1 (`http://jade.tilab.com`). Na Figura B.2, o *Analysis Agent* é uma *interface* de inicialização do *BioAgents*.

A utilização do *JADE* deve-se a diversos fatores, a saber: (i) ser distribuído como software livre sob licença *LGPL*; (ii) a linguagem de programação suportada ser *Java*, possibilitando boa portabilidade; (iii) as especificações de *JADE* serem compatíveis com o padrão *The Foundation of Intelligent Physical Agents-FIPA* [2], oferecendo uma biblioteca de classes de protocolos de interação padronizados e prontas para serem instanciadas ou estendidas; (iv) não apresentar necessidade de implementar a plataforma de agentes, as funcionalidades e a ontologia de gerenciamento de agentes, nem os mecanismos de transporte e *parsing* de mensagens; (v) oferecer um transporte eficiente de mensagens entre os agentes pelo uso da linguagem *FIPA Agent Communication Language - FIPA ACL* (`http://www.fipa.org/repository/aclspecs.html`); (vi) possuir suporte a usuários, tendo uma grande comunidade ativa de desenvolvedores e uma vasta documentação disponível para consulta.

Os *parsers* utilizados pelos agentes ANL para a manipulação dos arquivos de saída foram implementados a partir da adaptação de algumas bibliotecas do *framework BioJava* versão 1.4. O *BioJava* fornece objetos para manipulação

[2]FIPA é uma organização que segue o padrão internacional de especificação da *Institute of Electrical and Electronics Engineers - IEEE* para o desenvolvimento de tecnologias baseadas em agentes inteligentes de software (`http://www.fipa.org`).

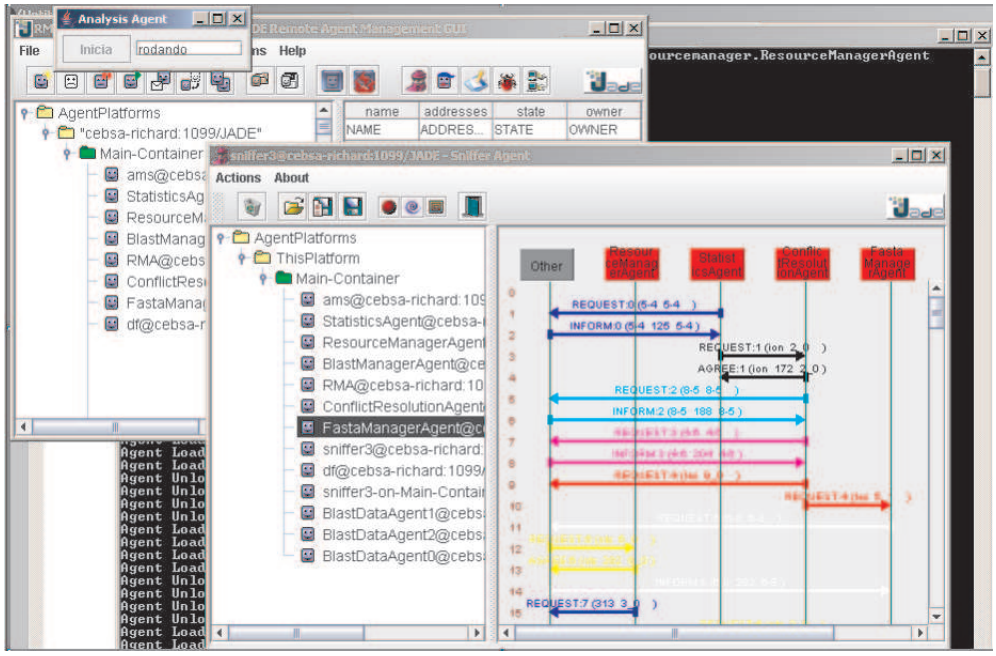


Figura B.2: *Screenshot* da tela de execução e do *sniffer* dos agentes do *BioAgents* no framework *JADE*.

de seqüências biológicas e *parsers* para arquivos de seqüências, dentre outras funcionalidades (`http://biojava.org/wiki/Main_Page`).

Como motor de inferência para o desenvolvimento do protótipo utilizamos o *Java Expert System Shell-JESS* versão 6.1 (`http://www.jessrules.com/jess/index.shtml`) [Friedman-Hill 2003]. O *JESS* é utilizado para construir bancos de conhecimento e obter inferências a partir de padrões pré-estabelecidos. O *JESS* foi especialmente desenvolvido para ser integrado à linguagem *Java*, o que permite a criação de *softwares Java* com capacidade de resolução de problemas usando conhecimento vindo das regras de produção implementadas no *JESS*. Estas regras representam o conhecimento explícito utilizado pelos biólogos na tarefa de anotação manual estando relacionadas ao conhecimento tácito utilizado durante o processo de sugestões de anotação.

4. Estudo de Caso

O estudo de caso realizado neste trabalho consistiu em utilizar o *BioAgents* com os dados do Projeto Genoma Pb, visando propor anotação a partir dos resultados *BLAST* e *FASTA* deste projeto, para comparar as anotações sugeridas com as anotações manuais previamente concluídas pelos biólogos. Os dados analisados foram os arquivos de saída do programa *BLAST* executado sobre os bancos *nr*, *COG* e *GO*; os arquivos de saída do programa *FASTA* com os bancos de dados dos fungos *Saccharomyces cereviseae* e *Schizosaccharomyces pombee*, bem como os arquivos de anotações manuais do Projeto Genoma Pb.

Para avaliar os arquivos de saída do *BLAST* e *FASTA*, o *BioAgents* analisou dois parâmetros, o *expectation-value (e-value)* e o *score*. Estes dois parâmetros

são produzidos pelo *BLAST* e pelo *FASTA* e expressam o grau de similaridade entre cada seqüência gerada no projeto e cada seqüência já existente em um banco de dados. Ambos os programas produzem *alinhamentos* entre duas seqüências, que expressam o grau de similaridade entre elas. Quanto menor o *e-value* maior a semelhança entre duas seqüências, e quanto maior o *score* mais próximas são as seqüências. A inferência de função é feita assumindo que quanto maior a proximidade entre duas seqüências, maior a chance de possuirem a mesma função biológica.

```
(defglobal ?*maxEvalue* = 1.0E-5); (1)

(defmodule Evalue)

(defrule Exists_Evalue_Above_Limit
    "Activate GoodEvalueAnalysis module if there is at least one good evalue"
    (exists (BlastHit
                (HitEvalue ?evalue&:(<= ?evalue ?*maxEvalue*)) (2)
    ))
    =>
    (focus GoodEvalueAnalysis)
    (run)
)

(defmodule GoodEvalueAnalysis)

(defrule Best_Evalue
 "comment"
 (EvalueAnalysis(Evalue ?evalue1)(Score ?score1))
 ?fact <- (EvalueAnalysis(Evalue ?evalue2&:(>= ?evalue2 ?evalue1))
                                          (Score ?score2&:(< ?score2 ?score1))) (3)
 =>
 (retract ?fact)
)
```

Figura B.3: Conjunto de regras *Jess* para análise de saídas *BLAST* e *FASTA*.

A Figura B.3 ilustra a sintaxe de duas regras com uso do *JESS*. Ressaltamos que estas regras foram testadas com os agentes GR e ANL, usando os programas *BLAST* e *FASTA*. As regras descritas nesta figura capturam o seguinte conhecimento biológico:

- Verificar a existência de alinhamentos cujo *e-value* seja menor ou igual a 10^{-5} (valor estabelecido pelos biólogos no Projeto Genoma Pb);
- Dentre os alinhamentos que atendem à restrição anterior, selecionar o menor *e-value*;
- Caso existam dois *e-values* iguais, selecionar o de maior *score*.

Como resultado da aplicação do *BioAgents*, foram analisados 6.107 seqüências do Projeto Genoma Pb (Tabela B.1). Deste total, 3.774 genes foram anotados manualmente por biólogos, e 2.333 não foram anotados. Na Tabela B.1 podemos observar um tempo de execução longo, motivado pelo fato dos dados do Projeto Genoma Pb serem compostos por arquivos do tipo texto com tamanho de aproximadamente 1.5 GB.

Note que 3.502 anotações foram sugeridas pelo *BioAgents*, sendo que 1.547 foram sugestões corretas quando comparadas com as anotações manuais do Projeto Genoma Pb, o que corresponde a um índice de acerto de 44.17%. Note também que das 1.955 sugestões não corretas quando comparadas com as anotações manuais do Projeto Genoma Pb, 336 foram sugestões do sistema a genes não anotados,

Tabela B.1: Resultados do *BioAgents* utilizando dados do Projeto Genoma Pb.

Quantidade de genes	6.107
Quantidade de genes anotados manualmente	3.774
Quantidade de anotações sugeridas pelo *BioAgentes*	3.502
Quantidade de anotações acertadas pelo *BioAgentes*/ Quantidade de anotações sugeridas(% de acerto)	1.547/3.502 44.17%
Quantidade de anotações sugeridas para genes não anotados manualmente/total de genes não anotados	336/2.333
Tempo de execução do sistema (hh:mm)	01:30

o que corresponde a 9.59% (336/3.502), e 1.619 foram sugestões diferentes das anotadas pelos biólogos, correspondendo a 46.23% (1.619/3.502). Conforme avaliação dos biólogos, os resultados são bons e podem ainda ser melhores à medida que for expandida a base de conhecimento dos agentes.

Com base nos resultados deste estudo de caso, julgamos que o *BioAgents* pode realmente auxiliar os biológos na fase de anotação manual em projetos de seqüenciamento de genomas.

5. Conclusões e Trabalhos Futuros

Neste trabalho, apresentamos uma arquitetura, baseada no paradigma Multiagente, e o protótipo do sistema *BioAgents* para apoiar o processo de anotação manual feita por biológos em projetos de seqüenciamento de genomas. Esta aplicação possui ambiente heterogêneo e dinâmico, pois utiliza diferentes bancos de dados, descentralizados, sendo os dados constantemente alterados. Assim, esta aplicação é adequada para ser solucionada utilizando a abordagem Multiagente. No *BioAgents* os agentes são especializados em tarefas distintas, de tal forma que podem atuar de forma independente, utilizando regras específicas. Esta arquitetura foi implementada utilizando o *framework JADE*, e as regras da base de conhecimento foram desenvolvidas no *JESS*.

Realizamos um estudo de caso com os dados de anotação manual do Projeto Genoma Pb. Usando poucas regras de produção, tivemos um índice de acerto de 44.17%, computado a partir do número de sugestões corretas do *BioAgents* quando comparadas com as anotações manuais do Projeto Genoma Pb. Além disso, o projeto sugeriu 336 anotações para seqüências não anotadas, consideradas corretas pelos biólogos que analisaram os dados.

Trabalhos futuros incluem a implementação com execução distribuída dos agentes, para reduzir o tempo de execução do *BioAgents*. Poderia ser desenvolvida uma *interface Web* para a *Camada de Apresentação*, provendo o acesso público aos pesquisadores que utilizassem o sistema. Pretendemos também utilizar o *BioAgents* no Projeto Genoma Anaplasma que em breve estará na fase de anotação manual (`http://dna.biomol.unb.br/ANA/`). O aprimoramento do conhecimento dos agentes GR e ANL também é necessário para possibilitar uma maior acurácia nas sugestões das anotações manuais. Isto poderia ser feito in-

cluindo novos métodos e bases de dados (como detecção de *RNAs* não-codificadores (*ncRNAs*), identificação de *RNAs* de transferência (*tRNAs*) e identificação de homologias em famílias de proteínas - *HMMER/Pfam*).

Referências

Altschul, S. F., Gish,W., Miller,W., Myers, E.W., and Lipmanl, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, pages 403410.

Bellifemine, F., Caire, G., Poggi, A., and Rimassa, G. (2003). Jade - a white paper. White Paper 3, TILAB - Telecom Italia Lab.

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., and Wheeler, D. L. (2006). Genbank. *Oxford Journals, Nucleic Acids Research*, 34:D16D20.

Decker, K., Zheng, X., and Schmidt, C. (2001). A multi-agent system for automated genomic annotation. In AGENTS 01: *Proceedings of the* 5^{th} *international conference on Autonomous agents*, New York, NY, USA. ACM Press.

Ding, L., Sabo, A., Berkowicz, N., Meyer, R. R., Shotland, Y., Johnson, M. R., Pepin, K. H., Wilson, R. K., and Spieth, J. (2004). Eannot: A genome annotation tool using experimental evidence. *Genome Research*, 14(12):25032509.

Felipe, M. S. S., Andrade, R. V., Arraes, F. B. M., Nicola, A. M., and et al (2005). Transcriptional profiles of the human pathogenic fungus paracoccidioides brasiliensis in mycelium and yeast cells. *Journal of Biological Chemistry (JBC)*, 280(26):24706 24714.

Friedman-Hill, E. (2003). *Jess in Action: Rule-Based Systems in Java*. Manning Publications Co, Greenwich, CT.

Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., and et al (2001). Initial sequencing and analysis of the human genome. *Nature*, 409:860921.

Lemos, M. (2004). *Workflow para bioinformática*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro (Puc-Rio).

Lima, R. S., Ralha, C. G., Walter, M. E. M. T., and Brígido, M. M. (2005). A multiagent system to help manual annotation on genome sequencing projects. Proceedings of the IGWD05 - International Workshop on Genomic Databases and Problem- Rio de Janeiro, Brazil, November 2005. Disponível em: `http://www.biowebdb.org/iwgd05/proceedings/multiagent-system.pdf`. Acesso em: Fevereiro de 2007.

Liolios, K., Tavernarakis, N., Hugenholtz, P., and Kyrpides, N. C. (2006). The genomes on line database (gold) v.2: a monitor of genome projects worldwide.

Oxford Journals, Nucleic Acids Research, 34:D332D334.

Nascimento, L. V. and Bazzan, A. L. (2005). An agent-based system for re-annotation of genomes. *Genetics and Molecular Research*, 4(3).

Pearson, W. R. and Lipman, D. J. (1988). Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the USA*, 85:24442448.

Santos, C. T. and Bazzan, A. L. C. (2004). Using the A3C system for annotation of keywords - a case study. *III Brazilian Workshop on Bioinformatics (WOB)*. Brasília, DF.

Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., and et al (2001). The sequence of the human project. *Science*, 291(16):13041351.

Watson, J. O. and Crick, F. H. C. (1953). Molecular structure of nucleic acids- a structure for deoxyribose nucleic acid. *Nature*, 171(4356):737738.

Weiss, G., editor (2000). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. The MIT Press, Cambridge, Massachusetts.

Wooldridge, M. (2002). *An Introduction to Multiagent Systems*. John Wiley Sons, LTD, England.

B.3 Terceira publicação

- BioAgents: A Multiagent System for Manual Annotation on Genome Sequencing Projects.
  - Autores: Richardson Silva Lima, Célia Ghedini Ralha, Maria Emilia Machado Telles Walter, Hugo Wruck Schneider, Anderson Gray Frazzon Pereira e Marcelo de Macedo Brígido.
  - Evento: International Workshop on Genomic Databases (IWGD).
  - Local e Data de Realização: Angra dos Reis, RJ, Brasil. 29 a 31 de agosto de 2007.
  - Tipo da publicação: Resumo expandido.
  - Idioma: Inglês.
  - Breve Resumo: Este trabalho apresentou o protótipo *BioAgents*, sua arquitetura, resultados obtidos com o estudo de caso relacionado ao Projeto Genoma Pb e Guaraná e alguns possíveis trabalhos futuros.

Íntegra do artigo:

The annotation phase on genome sequencing projects has the objective to assign biological functions to the DNA sequences obtained on the project. The annotation phase is divided into two tasks: (i) the automatic annotation task, that infers biological functions to each sequence, based on approximated comparison algorithms and databases containing sequences with corresponding functions; and (ii) the manual annotation task, in which the biologists guarantees accuracy and correctness to each sequence function. Thus, providing computational tools to assist the manual annotation task will certainly improve the final annotation. To achieve this, we take as hypothesis that the biologists decide the annotation of any sequence based in his knowledge, acquired from their professional experience.

In this context, this work presents BioAgents a system based on the multiagent approach to help the biologists with the manual annotation task on genome sequencing projects. BioAgents was developed under a blackboard multiagent architecture in JADE framework as shown in Figure B.4. The architecture is divided into three layers: interface, collaborative and physical. The interface layer receives the requests and returns the results to users. The collaborative layer is the architecture core and has agents that interact with manager agents specialized in BLAST and FASTA, local databases and different knowledge sources to suggest annotations to be sent to the interface layer. The physical layer consists of different local databases containing the results of the automatic annotation. Note that specialized agents in distinct tasks interact with local databases and specific knowledge sources (KS).

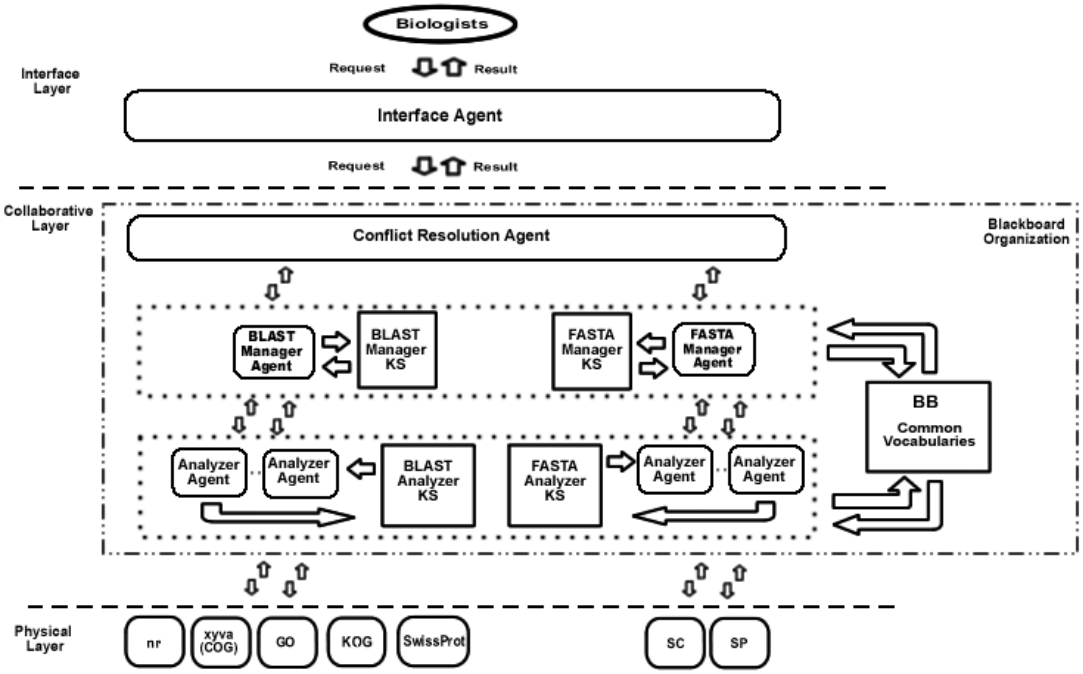


Figura B.4: BioAgents Architecture.

In order to validate BioAgents, we used data from the manual annotations of two genome projects: (i) the Paraccocidiodes brasiliensis (Genome Pb Project)

and the Paullinia Cupana (Genome Guarana Project). The knowledge of the agents was represented by production rules, which were used to infer the suggested annotations which will be later validated by the biologists. The production rules implemented at the expertise module have used only two parameters: (i) e-value less than or equal to 10-5 and (ii) in case of the same e-value the greater score. The Genome Pb Project has 6,107 sequences with 3,780 manually annotated. BioAgents suggested 3,496 annotations, where 1,547 were correct when compared to the previous manual annotation, corresponding to 44.25% of correctness. While for the Genome Guarana Project, with 8,597 sequences and 7,725 manually annotated, BioAgents suggested 6,478 annotations, where 2,938 were correct, corresponding to 45.35% of correcteness. The experiments suggests that BioAgents can really help biologists during the manual annotation phase of a genome sequencing project since we had more than 40% of correctness. As future work we intend to improve the expertise of the system in order to obtain better suggestions. We also intend to use BioAgents at not manually annotated genome sequencing projects to accelerate the anotation fase. This will be done for the manual annotation task of the Anaplasma marginale genome sequencing project.

References

1. Altschul SF, et al: **Basic Local Alignment Search Tool**. *Journal of Molecular Biology* 1990, 215(3): 403-410.

2. Andrade MA, et al: **Automated genome sequence analysis and annotation**. *Bioinformatics* 1999, 15(5): 391-412.

3. Bazzan ALC, et al: **ATUCG - An Agent-Based Environment for Automatic Annotation of Genomes**. *International Journal of Cooperative Information Systems* 2003, 12(2): 241-273.

4. Decker K, et al: **BioMAS: A Multi-Agent System for Genomic Annotation**. *International Journal of Cooperative Information Systems* 2002, 11(3): 265-292.

5. Ding Li, et al: **EAnnot: A genome annotation tool using experimental evidence**. *Genome Research* 2004, 14(12): 2503-2509.

6. Genome *Anaplasma marginale* Project [`http://www.biomol.unb.br/anaplasma/servlet/IndexServlet`].

7. Genome Guaraná Project [`http://dna.biomol.unb.br/GR/`].

8. Genome Pb Project [`http://dna.biomol.unb.br/Pb/`].

9. Hill EF: **Jess in Action: Rule-Based Systems in Java**. Maning Publications Co; 2003.

10. JADE-Java Agent Development Framework [http://jade.tilab.com/].

11. Nascimento LV, Bazzan ALC: **An agent-based system for re-annotation of genomes**. *Genetics and Molecular Research* 2005, 4(3): 571-580.

12. Pearson WR and Lipman DJ: **Improved tools for biological sequence comparison**. *PNAS* 1988, 85(8): 2444-2448.

13. Weiss G: **Multiagent Systems - A Modern Approach to Distributed Modern Approach to Artificial Intelligence**. MIT Press; 2000.

14. Wooldridge M: **An Introduction to MultiAgent Systems**. John Wiley Sons Ltd; 2002.

Referências

[1] http://www.ktf-split.hr/glossary.

[2] http://www.humboldt.edu/~rap1.

[3] http://www.genelex.com/paternitytesting.

[4] http://www.genomesonline.org.

[5] http://www.ebi.ac.uk.

[6] http://pfam.wustl.edu.

[7] http://www.tigr.org.

[8] http://www.sanger.ac.uk.

[9] http://www.ncbi.nlm.nih.gov.

[10] http://www.biomol.unb.br.

[11] http://www.biofoco.org.

[12] http://www.uml.org.

[13] http://transterm.cbcb.umd.edu.

[14] http://www.ncbi.nlm.nih.gov/blast.

[15] http://fasta.bioch.virginia.edu.

[16] http://www.yeastgenome.org.

[17] http://www.genedb.org/genedb/pombe.

[18] http://www.expasy.org/sprot.

[19] http://www.psort.org.

[20] http://selab.janelia.org.

[21] http://www.ddbj.nig.ac.jp.

[22] http://arep.med.harvard.edu/seqanal/db.html.

[23] http://www.geneontology.org.

[24] http://www.ebi.ac.uk/trembl.

[25] http://www.sanger.ac.uk/Software/Rfam.

[26] http://smart.embl-heidelberg.de.

[27] http://java.sun.com.

[28] http://jade.tilab.com.

[29] http://labs.bt.com/projects/agents/zeus.

[30] http://www.cs.umbc.edu/kqml.

[31] http://www.fipa.org.

[32] http://www.ieee.org.

[33] http://www.gnu.org/licenses/lgpl.html.

[34] http://www.mygrid.org.uk.

[35] http://protege.stanford.edu/.

[36] http://www.postgresql.org/.

[37] http://jura.ebi.ac.uk:8765/ext-genequiz.

[38] http://www.cis.udel.edu/~decaf.

[39] http://www.cbcb.umd.edu/software/glimmer.

[40] http://exon.gatech.edu/GeneMark.

[41] Márjory Cristiany Costa Abreu. Analisando o desempenho do ClassAge: Um sistema multiagentes para classificação de padrões. Master's thesis, Universidade Federal do Rio Grande do Norte (UFRN), 2006. Disponível em: http://www.ppgsc.ufrn.br/html/Producao/Dissertacoes/MarjoryCristianyDaCostaAbreu.pdf. Acessado em: Abril de 2007.

[42] Luiz G. P. Almeida, Roger Paixao, Rangel C. Souza, Gisele C. da Costa, and et al. A system for automated bacterial (genome) integrated annotation – SABIA. *Oxford Journals, Bioinformatics*, 20(16):2832–2833, 2004.

[43] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

[44] José Mariano Amabis and Gilberto Rodrigues Martho. *Fundamentos da Biologia Moderna*. Editora Moderna, São Paulo, SP, Brasil, 1997.

[45] Miguel A. Andrade, Nigel P. Brown, Christophe Leroy, Sebastian Hoersch, and et al. Automated genome sequence analysis and annotation. *Oxford Journals, Bioinformatics*, 15(5):391–412, 1999.

[46] Luis Alberto Santos Antunes. *Agentes com Decisão Baseada em Valores*. PhD thesis, Universidade de Lisboa, Lisboa, Portugal, 2001. Disponível em: `http://www.di.fc.ul.pt/~xarax/publications/antunes-phd01.pdf`. Acessado em: Abril de 2007.

[47] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, and et al. Gene Ontology: tool for the unification of biology. *Nature*, 25:25–29, 2000.

[48] Amos Bairoch, Lydie Bougueleret, Severine Altairac, Valeria Amendolia, Andrea Auchincloss, and et al. The universal protein resource (uniprot). *Oxford Journals, Nucleic Acids Research*, 35:D193–D197, 2007. Disponível em: `http://nar.oxfordjournals.org/cgi/reprint/35/suppl_1/D193`. Acessado em: Junho de 2007.

[49] Patricia G. Baker, Andy Brass, Sean Bechhofer, Carole Goble, Norman Paton, and Robert Stevens. TAMBIS: Transparent access to multiple bioinformatics information sources. In *Proceedings of the Sixth International Conference on Intelligent Systems for Molecular Biology*, Montreal, Canada, June 1998. Disponível em: `http://citeseer.ist.psu.edu/baker98tambis.html`. Acessado em: Abril de 2007.

[50] Patricia G. Baker, Carole A. Goble, Sean Bechhofer, Norman W. Paton, Robert Stevens, and Andy Brass. An ontology for bioinformatics applications. *Oxford Journals, Bioinformatics*, 15(6):510–520, 1999.

[51] Alex Bateman, Lachlan Coin, Richard Durbin, Robert D. Finn, Volker Hollich, Sam Griffiths-Jones, Ajay Khanna, Mhairi Marshall, Simon Moxon, Erik L. L. Sonnhammer, David J. Studholme, Corin Yeats, and Sean R. Eddy. The Pfam protein families database. *Nucleic Acids Research, Oxford Journals*, pages 138–141.

[52] Christian Baudet. Uma abordagem para trimagem, verificação de contaminação e clusterização de seqüências EST. Master's thesis, Universidade Estadual de Campinas (Unicamp), 2006. Disponível em: `http://www.ic.unicamp.br/~zanoni/orientandos/christian/TeseChristian.pdf`. Acessado em: Maio de 2006.

[53] Andreas D. Baxevanis and B. F. Francis Ouellette. *BIOINFORMATICS: A Practical Guide to the Analysis of Genes and Proteins*. John Wiley Sons, Inc, New York, USA, 2001.

[54] Ana L. C. Bazzan, Rogério Duarte, Abner N. Pitinga, Luciana F. Schroeder, Farlon de A. Souto, and Sérgio Ceroni da Silva. ATUCG – An Agent-Based Environment for a Automatic Annotation of Genomes. *International Journal of Cooperative Information Systems (IJCIS)*, 12(2):241–273, 2003.

[55] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa. *JADE - A White Paper*. TILAB - Telecom Italia Lab, September 2003. Disponível em: `http://jade.tilab.com/papers/2003/WhitePaperJADEEXP.pdf`. Acessado em: Dezembro de 2006.

[56] Fabio Bellifemine, Giovanni Caire, Tiziana Trucco, and Giovanni Rimassa. *JADE Programmers Guide*. TILAB, Italia, November 2005. Disponível em: `http://jade.tilab.com/doc/programmersguide.pdf`. Acessado em: Junho de 2007.

[57] Dennis A. Benson, Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and David L. Wheeler. Genbank. *Oxford Journals, Nucleic Acids Research*, 34:D16–D20, 2006.

[58] William P. Birmingham. Essential issues in distributed computational systems. NSF Invitational Workshop on Distributed Information, Computation, and Process Management for Scientific and Engineering Environments (DICPM), May 1998. Disponível em: `http://deslab.mit.edu/DesignLab/dicpm/position/birmingham.html`. Acessado em: Abril de 2007.

[59] Brigitte Boeckmann, Amos Bairoch, Rolf Apweiler, Marie-Claude Blatter, Anne Estreicher, Elisabeth Gasteiger, Maria J. Martin, Karine Michoud, Claire ODonovan, Isabelle Phan, Sandrine Pilbout, and Michel Schneider. The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Oxford Journals, Nucleic Acids Research*, 31(1):365–370, 2003. Disponível em: `http://nar.oxfordjournals.org/cgi/reprint/31/1/365/`. Acessado em: Novembro de 2006.

[60] Eduardo Sette Brüggemann and Ricardo Marques Porto. Uma proposta ontológica para um sistema de gestão de versionamento do modelo ITIL. Graduation Work at University of Brasília (UnB), March 2002.

[61] Marcelo M. Brígido, Maria Emília M.T. Walter, Adilton G. Oliveira, Marcus K. Inoue, and et al. Bioinformatics of the Paracoccidioides brasiliensis EST Project. *Genetics and Molecular Research*, 4(2):203–215, 2005.

[62] Terence A. Brown. *Genomes*. BIOS Scientific Publishers Ltd, Oxford, UK, 2002.

[63] Giovanni Caire and David Cabanillas. *Application-Defined Content Languages and Ontologies*. TILAB, Italia, November 2004.

[64] Alessandra C. Farias Campos, Daniela V. Campos Barbosa, Francisco Lobo, and Raquel Cardoso de Melo. Apostila de bioinformática, 2006. Disponível em: `http://biotec.icb.ufmg.br/cabi/apostila_completa.pdf`. Acessado em: Abril de 2007.

[65] Leila Fátima Sousa Carvalho. BLOOM - Blast Object Oriented Management: uma solução integrada para gerenciamento dos resultados do

blast por meio de um paradigma orientado a objetos. Master's thesis, Universidade Católica de Brasília (UCB), 2002. Disponível em: `http://www.biofoco.org/wedo/bloom.pdf`. Acessado em: Março de 2007.

[66] Ricardo Celestino. *FIPA-OS*, 2006. Disponível em: `http://www.larces.uece.br/tutoriais/FIPAOS_RICARDO_TUTORIAL.PDF`. Acessado em: Maio de 2007.

[67] Guy Cochrane, Philippe Aldebert, Nicola Althorpe, Mikael Andersson, and et al. Embl nucleotide sequence database: developments in 2005. *Oxford Journals, Nucleic Acids Research*, 34:D10–D15, 2006.

[68] John Cohen. Bioinformatics-an introduction for computer scientists. *ACM Computing Surveys*, 36(2):122–158, 2004.

[69] Rodrigo Carneiro Munhoz Coimbra and Shana Schlottfeldt Santos. Projeto, implementação e aplicação de um framework de código aberto para projetos de seqüenciamento de genomas. Graduation Work at University of Brasília (UnB), August 2006.

[70] Daniel D. Corkill. Blackboard systems. *AI Expert*, 6(9):40–47, September 1991.

[71] Daniel D. Corkill. Collaborating software: Blackboard and multi-agent systems & the future. In *Proceedings of the International Lisp Conference*, New York, USA, October 2003. Disponível em: `http://mas.cs.umass.edu/paper/265`. Acessado em: Março de 2007.

[72] David De Roure, M. Baker, N. R. Jennings, and N. Shadbol. The evolution of the grid. In F. Berman, G. Fox, and A. J. G. Hey, editors, *Grid Computing: Making the Global Infrastructure a Reality*, Series in Communications Networking and Distributed Systems, chapter 3, pages 65–100. Wiley, 2003. Disponível em: `http://sorma.fzi.de/images/d/dd/Rou03.pdf`. Acessado em: Abril de 2007.

[73] K. Decker, S. Khan, C. Schmidt, G. Situ, R. Makkena, and D. Michaud. BioMAS: A multi-agent system for genomic annotation. *International Journal of Cooperative Information Systems (IJCIS)*, 11(3):265–292, 2002.

[74] Sarah E. DeWeerdt, Barbara J. Culliton, and Mary S. Gibbs. What's a genome? Technical report, Genome News Network, 2003. Disponível em: `http://www.genomenewsnetwork.org/resources/whats_a_genome/`. Acessado em: Maio de 2007.

[75] Li Ding, Aniko Sabo, Nicolas Berkowicz, Rekha R. Meyer, Yoram Shotland, Mark R. Johnson, Kymberlie H. Pepin, Richard K. Wilson, and John Spieth. EAnnot: A genome annotation tool using experimental evidence. *Genome Research*, 14(12):2503–2509, 2004.

[76] Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme J. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge, UK, 1998.

[77] A. M. R. Dávila, D. M. Lorenzini, P. N. Mendes, T. S. Satake, G. R. Sousa, L. N. Campos, C. J. Mazzoni, G. Wagner, P. F. Pires, E. C. Grisard, M. C. R. Cavalcanti, and M. L. M. Campos. Garsa: genomic analysis resources for sequence annotation. *Oxford Journals, Bioinformatics*, 21(23):4302–4303, 2005.

[78] S. Eddy. A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an rna secondary structure. *BMC Bioinformatics*, 3(18), 2002.

[79] Sean Eddy. INFERNAL user's guide: Sequence analysis using profiles of rna secondary structure consensus. Technical report, Howard Hughes Medical Institute and Dept. of Genetics, Saint Louis, USA, 2007. Disponível em: `ftp://selab.janelia.org/pub/software/infernal/Userguide.pdf`. Acessado em: Fevereiro de 2007.

[80] Sean R. Eddy. Profile hidden markov models. *Oxford Journals, Bioinformatics*, 14(9):755–763, 1998.

[81] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37–54, 1996.

[82] M. S. S. Felipe, R. V. Andrade, S. S. Petrofeza, A. Q. Maranhão, and et al. Transcriptome characterization of the dimorphic and pathogenic fungus paracoccidioides brasiliensis by est analysis. *Yeast*, 20(3):263–271, 2003.

[83] Maria Sueli S. Felipe, Rosângela V. Andrade, Fabrício B. M. Arraes, André M. Nicola, and et al. Transcriptional profiles of the human pathogenic fungus paracoccidioides brasiliensis in mycelium and yeast cells. *Journal of Biological Chemistry (JBC)*, 280(26):24706–24714, July 2005.

[84] J. Ferber and L. Gasser. Intelligence artificielle distribuée. In *Tutorial Notes of the 11th Conference on Expert Systems and their Applications.*, Avignon, France, May 1991.

[85] Jacques Ferber. *Multi-Agent System: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, Massachusetts, USA, 1999.

[86] Marcos Francisco Ribeiro Ferreira. Visualização de dados genômicos do fungo paracoccidioides brasiliensis. Master's thesis, Universidade de Brasília (UnB), 2006. Disponível em: `http://monografias.cic.unb.br/dspace/`. Acessado em: Maio de 2007.

[87] Stan Franklin and Art Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. In *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages. Spring-Verlag*, volume 1193, Berlin, Germany, 1996. Springer-Verlag. Disponível em: `http://citeseer.ist.psu.edu/franklin96is.html`. Acessado em: Março de 2007.

[88] C. Gibas and P. Jambeck. *Developing Bioinformatics Computer Skills*. O'Reilly, Sebastopol,CA, 2001.

[89] Eduardo Rodrigues Gomes. Objetos inteligentes de aprendizagem: uma abordagem baseada em agentes para objetos de aprendizagem. Master's thesis, Universidade Federal do Rio Grande do Sul (UFRGS), 2005. Disponível em: `http://ifm.ufpel.edu.br/iate/downloads/msc_egomes.pdf`. Acessado em: Maio de 2007.

[90] Artiva Maria Goudel. Criação da base de dados via/genoma da chromobacterium violaceum – cviocyc e análise das informações geradas pelo software pathway tools. Master's thesis, Universidade Federal de Santa Catarina (UFSC), 2005. Disponível em: `http://www2.enq.ufsc.br/teses/m143.pdf`. Acessado em: Abril de 2007.

[91] Sam Griffiths-Jones, Simon Moxon, Mhairi Marshall, Ajay Khanna, Sean R. Eddy, and Alex Bateman. Rfam: annotating non-coding rnas in complete genomes. *Oxford Journals, Nucleic Acids Research*, 33:D121–D124, 2005.

[92] Thomas R. Gruber. Towards principles for the design of ontologies used for knowledge sharing. In *Proceedings of International Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands, 1993. Kluwer Academic Publishers. Disponível em: `http://citeseer.ist.psu.edu/gruber93toward.html`. Acessado em: Abril de 2007.

[93] Ernest Friedman Hill. *Jess in Action: Rule-Based Systems in Java*. Manning Publications Co, Connecticut, USA, 2003.

[94] Hui-Huang Hsu. *Advanced Data Mining Technologies in Bioinformatics*. Idea Group Publishing, Pennsylvania, USA, 2006.

[95] Hui-Huang Hsu. *Introduction to Data Mining in Bioinformatics*, chapter 1, pages 1–12. Idea Group Publishing, 2006.

[96] Ronald Jansen, Ning Lan, Jiang Qian, and Mark Gerstein. Integration of genomic datasets to predict protein complexes in yeast. *Journal of Structural and Functional Genomics*, 2(2):71–81, 2002.

[97] Minoru Kanehisa, Susumu Goto, Masahiro Hattori, Kiyoko F. Aoki-Kinoshita, Masumi Itoh, Shuichi Kawashima, Toshiaki Katayama, Michihiro Araki, and Mika Hirakawa. From genomics to chemical genomics: new developments in kegg. *Oxford Journals, Nucleic Acids Research*, 34:D354–D357, 2006.

[98] Anders Krogh, Michael Brown, I. Saira Mian, Kimmen Sjölander, and David Haussler. Hidden markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235(5):1501–1531, 1994.

[99] Eric S. Lander, Lauren M. Linton, Bruce Birren, Chad Nusbaum, and et al. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, February 2001.

[100] Albert L. Lehninger, David L. Nelson, and Michael M. Cox. *Princípios de bioquímica*. Sarvier, São Paulo, SP, Brasil, 1995.

[101] Melissa Lemos. *Workflow para bioinformática*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro (Puc-Rio), 2004. Disponível em: `www.inf.puc-rio.br/~melissa/publicacao/download/tese_melissa/Tese_Melissa_Lemos.pdf`. Acessado em: Maio de 2007.

[102] Melissa Lemos, Luiz Fernando Bessa Seibel, and Marco Antônio Casanova. Sistemas de anotações em bioseqüências. Informática, PUC-Rio, Fevereiro 2003. Disponível em: `http://www.inf.puc-rio.br/~melissa/publicacao/download/mcc_melissa/MCC04-03.pdf`. Acessado em: Maio de 2007.

[103] Ivica Letunic, Richard R. Copley, Birgit Pils, Stefan Pinkert, Jörg Schultz, and Peer Bork. Smart 5: domains in the context of genomes and networks. *Oxford Journals, Nucleic Acids Research*, 34:D257–D260, 2006.

[104] Li Liao. *Hierarchical Profiling, Scoring and Applications in Bioinformatics*, chapter 2, pages 13–31. Idea Group Publishing, advanced data mining technologies in bioinformatics. first edition, 2006.

[105] Richardson S. Lima, Célia G. Ralha, Maria Emília M. T. Walter, and Marcelo M. Brígido. A Multiagent System to Help Manual Annotation on Genome Sequencing Projects. In *Proceedings of the International Workshop on Genomic Databases and Problem (IWGD)*, Rio de Janeiro, RJ, Brasil, November 2005. Disponível em: `http://www.biowebdb.org/iwgd05/proceedings/multiagent-system.pdf`. Acessado em: Maio de 2007.

[106] Richardson S. Lima, Célia G. Ralha, Maria Emília M. T. Walter, Hugo W. Schneider, Anderson G. F. Pereira, and Marcelo M. Brígido. BioAgents: A Multiagent System for Manual Annotation on Genome Sequencing Projects. In *Proceedings of the International Workshop on Genomic Databases and Problem (IWGD)*, Angra dos Reis, RJ, Brasil, August 2007.

[107] Richardson S. Lima, Célia G. Ralha, Maria Emília M. T. Walter, Hugo W. Schneider, Anderson G. F. Pereira, and Marcelo M. Brígido. BioAgents: Um Sistema Multiagente para Anotação Manual em Projetos de Seqüenciamento de Genomas. *VI Encontro Nacional de Inteligência Artificial (ENIA). Anais do XXVII Congresso da Sociedade Brasileira de Computação (CSBC)*, 2007.

[108] Konstantinos Liolios, Nektarios Tavernarakis, Philip Hugenholtz, and Nikos C. Kyrpides. The genomes on line database (gold) v.2: a monitor of genome projects worldwide. *Oxford Journals, Nucleic Acids Research*, 34:D332–D334, 2006.

[109] D. J. Lipman and W. R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, March 1985.

[110] Todd Lowe. *tRNAscan-SE: a program for improved transfer RNA detection in genomic sequence*. University of California, California, USA, 2001. Disponível em: `ftp://selab.janelia.org/pub/software/tRNAscan-SE/tRNAscan-SE.tar.Z`. Acessado em: Abril de 2007.

[111] Todd M. Lowe and Sean R. Eddy. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Oxford Journals, Nucleic Acids Research*, 25(5):955–964, 1997.

[112] George F. Luger. *Inteligência Artificial: estruturas e estratégias para a solução de problemas complexos*. Editora Bookman, Porto Alegre, RS, Brasil, 2004.

[113] Pattie Maes. Artificial life meets entertainment: lifelike autonomous agents. *ACM*, 38(11):108–114, 1995.

[114] Scott McGinnis and Thomas L. Madden. Blast: at the core of a powerful and diverse set of sequence analysis tools. *Oxford Journals, Nucleic Acids Research*, 32:W20–W25, 2004.

[115] Folker Meyer, Alexander Goesmann, A. C. McHardy, T. Bekel D. Bartels, J. Clausen, J. Kalinowski, B. Linke, O. Rupp, R. Giegerich, and A. Pühler. Gendb - an open source genome annotation system for prokaryote genomes. *Oxford Journals, Nucleic Acids Research*, 31(8):2187–2195, 2003.

[116] Eugene W. Myers and Webb Miller. Optimal alignments in linear space. *Oxford Journals, Bioinformatics*, 4(1):11–17, 1988.

[117] Kenta Nakai and Paul Horton. Psort: a program for detecting sorting signals in proteins and predicting their subcellular localization. *Trends in Biochemical Sciences*, 24(1):34–35, January 1999.

[118] Leonardo Vianna Nascimento and Ana L.C. Bazzan. An agent-based system for re-annotation of genomes. *Genetics and Molecular Research*, 4(3):571–580, 2005.

[119] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, March 1970.

[120] A. L. Nepomuceno, J. F. V. Silva, E. G. M. Lemos, E. Binneck, and et al. Genoma funcional de raízes de soja, 2003. Disponível em: `http://www.cnpso.embrapa.br/bioinformatica/explorer/`. Acessado em: Abril de 2007.

[121] H. P. Nii. Blackboard systems, part one: The blackboard model of problem solving and the evolution of blackboard architectures. *AI Magazine*, 7(2):38–53, 1986.

[122] Fabio Yoshimitsu Okuyama. Descrição e geração de ambientes para simulações com sistemas multiagentes. Master's thesis, Universidade Federal do Rio Grande do Sul (UFRGS), 2003. Disponível em: `www.inf.ufrgs.br/~okuyama/dissOkuyama.pdf`. Acessado em: Abril de 2007.

[123] A.G. Oliveira and M.K. Inoue. Laboratório de bioinformática do projeto genoma funcional e diferencial do fungo *Paracoccidioides brasiliensis* - projeto genoma pb. Graduation Work at University of Brasília (UnB), September 2002.

[124] William R. Pearson and David J. Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the USA (PNAS)*, 85(8):2444–2448, April 1988.

[125] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[126] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):257–286, February 1989.

[127] Tania G. Ramos. Mecanismo extensível de descoberta de recursos em ambiente de computação em grade. Master's thesis, Universidade de Brasília (UnB), 2005.

[128] Luis Paulo Reis. *Coordenação em Sistemas Multi-Agente: Aplicações na Gestão Universitária e Futebol Robótico*. PhD thesis, Faculdade de Engenharia da Universidade do Porto (FEUP), 2003. Disponível em: `http://paginas.fe.up.pt/~lpreis/`. Acessado em: Maio de 2007.

[129] S. Russell and P. Norvig. *Inteligência Artificial*. Editora Campus, Rio de Janeiro, RJ, Brasil, 2004.

[130] Cassia T. Santos and Ana L. C. Bazzan. Integrating Knowledge through Cooperative Negotiation – A Case Study in Bioinformatics. In *Proceedings of the International Workshop on Autonomous Intelligent Systems: Agents and Data Mining*, number 3505, pages 277–288, St. Petersburg, Russia, June 2005. Springer-Verlag. Disponível em: `http://www.inf.ufrgs.br/bazzan/downloads/AISDM3505_277.pdf.gz`. Acessado em: Junho de 2007.

[131] Eluzaí Souza Santos. Uma proposta de integração de sistemas computacionais utilizando ontologias. Master's thesis, Universidade de Brasília (UnB), 2006. Disponível em: `http://monografias.cic.unb.br/dspace/`. Acessado em: Maio de 2007.

[132] Gregory G. Schuler. *Sequence Alignment and Data Base Searching*, chapter 8, pages 187–214. John Wiley Sons, Inc, bioinformatics: a practical guide to the analysis of genes and proteins. second edition, 2001.

[133] Jörg Schultz, Frank Milpetz, Peer Bork, and Chris P. Ponting. SMART, a simple modular architecture research tool: Identification of signaling domains. *Proceedings of The National Academy of Sciences of the USA (PNAS)*, 95(11):5857–5864, May 1998.

[134] J. C. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology*. PWS Publishing, Massachusetts, USA, 1997.

[135] João Carlos Setubal. Notas de aula de biologia computacional, 2003. Disponível em: `http://onsona.lbi.ic.unicamp.br/biocomp/`. Acessado em: Maio de 2007.

[136] Flávio Henrique Silva. Módulo: Biologia molecular. I Escola de Inteligência Artificial e Bioinformática (InBio)- São Carlos-UFScar, Dezembro 2001. Disponível em: `http://www.ufscar.br/~dge/apostila_biomol_2001.pdf`. Acessado em: Maio de 2007.

[137] A. J. G. Simpson, F.C. Reinach, P. Arruda, F. A. Abreu, and et al. The genome sequence of the plant pathogen xylella fastidiosa - the xylella fastidiosa consortium of the organization for nucleotide sequencing and analysis. *Nature*, 406(6792):151–157, January 2000.

[138] Mona Singh. Profile hidden markov models. Technical report, Princeton University, 1999. Disponível em: `http://www.cs.princeton.edu/~mona/Lecture/`. Acessado em: Março de 2007.

[139] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.

[140] Marcelo Silva Sousa. Packageblast: Serviço de grade adaptativo com múltiplas políticas de alocação de tarefas para comparação de seqüências biológicas. Master's thesis, Universidade de Brasília (UnB), 2005.

[141] Hideaki Sugawara, Takashi Abe, Takashi Gojobori, and Yoshio Tateno. DDBJ working on evaluation and classification of bacterial genes in INSDC. *Oxford Journals, Nucleic Acids Research*, 35:D13–D15, 2007.

[142] Baris E. Suzek, Maria D. Ermolaeva, Mark Schreiber, and Steven L. Salzberg. A probabilistic method for identifying start codons in bacterial genomes. *Oxford Journals, Bioinformatics*, 17(12):1123–1130, 2001.

[143] K. Sycara. Multiagent systems. *AI Magazine*, 19(2):79–92, 1998.

[144] Katia Sycara, Massimo Paolucci, Anupriya Ankolekar, and Naveen Srinivasan. Automated discovery, interaction and composition of semantic web services. *Journal of Web Semantics*, 1(1):27–46, 2003.

[145] Andreas L. Symeonidis and Pericles A. Mitkas. *Agent Intelligence Through Data Mining*. Springer, New York, USA, 2006.

[146] Roman L. Tatusov, Natalie D. Fedorova, John D. Jackson, Aviva R. Jacobs, Boris Kiryutin, and et al. The cog database: an updated version includes eukaryotes. *BMC Bioinformatics*, 4(41), 2003.

[147] Roman L. Tatusov, Michael Y. Galperin, Darren A. Natale, and Eugene V. Koonin. The cog database: a tool for genome-scale analysis of protein functions and evolution. *Oxford Journals, Nucleic Acids Research*, 28(1):33–36, 2000.

[148] Alfonso Valencia. Automatic annotation of protein function. *Current Opinion in Structural Biology*, 15(3):267–274, 2005.

[149] Ana Tereza Ribeiro Vasconcelos, Darcy F. de Almeida, Mariangela Hungria, Claudia Teixeira Guimarães, and et al. The complete genome sequence of chromobacterium violaceum reveals remarkable and exploitable bacterial adaptability. *Proceedings of the National Academy of Sciences of the USA (PNAS)*, 100(20):11660–11665, September 2003.

[150] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, and et al. The sequence of the human project. *Science*, 291(16):1304–1351, 2001.

[151] James O. Watson and Francis H. C. Crick. Molecular structure of nucleic acids- a structure for deoxyribose nucleic acid. *Nature*, 171(4356):737–738, April 1953.

[152] Gerhard Weiss. *Multiagent Systems - A Modern Approach to Distributed Modern Approach to Artificial Intelligence*. MIT Press, Massachusetts, USA, 2000.

[153] Danny Weyns, Andrea Omicini, and James Odell. Environment as a first-class abstraction in multiagent systems. *Special Issue on Environments for Multiagent Systems of Journal on Autonomous Agents and Multiagent Systems*, 14(1):5–30, 2006.

[154] Danny Weyns, Michael Schumacher, Alessandro Ricci, Mirko Viroli, and Tom Holvoet. Environments for multiagent systems. *The Knowledge Engineering Review*, 20(2):127–141, 2005.

[155] W. J. Wilbur and David J. Lipman. Rapid similarity searches of nucleic acid and protein data banks. *Proceedings of the National Academy of Sciences of the USA (PNAS)*, 80(3):726–730, February 1983.

[156] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers, California, USA, 2005.

[157] M. Wooldridge. *An Introduction to Multiagent Systems*. John Wiley Sons, LTD, West Sussex,England, 2002.

[158] Franco Zambonelli, Nicholas R. Jennings, and Michael Wooldridge. Developing multiagent systems: The gaia methodology. *ACM Transactions on Software Engineering and Methodology*, 12(3):317–370, 2003.

Glossário

Aminoácidos
Um proteína é composta por cadeias de centenas ou milhares de aminoácidos. Na quimíca, são definidos como compostos orgânicos que contenham o grupo (NH_2) e carboxil ($COOH$).

Bases Nitrogenadas
Base nitrogenada é um composto cíclico contendo nitrogênio. São cinco bases desse tipo, divididas em purinas - Adenina (A) e Guanina (G) - e pirimidinas - Uracila (U), Citosina (C) e Timina (T).

Códon É uma seqüência de três bases nitrogenadas que especificam uma aminoácido. Os códons de finalização indicam a finalização da síntese proteíca (não codificam nenhum aminoácido).

CAP3 É um *software* que realiza o processo de montagem das seqüências gerando um grupo de *contigs* e outros de *singlets*. O *software* agrupa seqüências com certas características comuns. Cada agrupamento gera uma única seqüência, o consenso.

Contigs São seqüências (consenso) formadas a partir do agrupamento de duas ou mais seqüências.

DNA complementar
O DNA complentar (*cDNA*) é sintetizado do *mRNA*. Pode ser usado experimentalmente para determinar a seqüência de um *mRNA*.

DNA É uma seqüência linear de quatro nucleotídeos (também chamados de bases nitrogenadas, representados pelos caracteres **A**, **T**, **C** e **G**, observando que a **A**denina se pareia com **T**imina e **G**uanina com **C**itosina), que é a fonte básica da informação genética. Esta informação é copiada ou transcrita para moléculas de *RNA*, cujas sequências de nucleotídeos contêm o código para a ordenação específica de uma sequência de aminoácidos (também chamados de resíduos, representados por 20 caracteres). As proteínas, que são sequências de aminoácidos, são então sintetizadas num processo que envolve a tradução do *RNA*.

Estrutura Secundária de uma proteína
É dada pelo arranjo espacial de aminoácidos próximos entre si na seqüência primária da proteína.

Estrutura Terciária de uma proteína
Descreve o dobramento final de uma cadeia, por interações de regiões com estrutura regular ou de regiões sem estrutura definida. Esta estrutura confere a atividade biológica às proteínas.

Exons São regiões codificadoras do gene, ou seja, são regiões do gene que determinam a seqüência de aminoácidos de uma proteína.

Filogenia Trata-se do estudo da história evolucionária dos organismos.

Fitopatógeno
Diz-se de organismos (insetos, fungos, bactérias, etc) capazes de produzir danos ou doenças em plantas.

Genes São pequenas partes de um cromossomo que contêm a informações necessárias para produzir um produto funcional, geralmente uma proteína.

Genoma É toda a informação hereditária do organismo que está codificada no seu *DNA* (ou, em alguns vírus, no *RNA*). Isto inclui tanto os genes como as sequências não-codificadoras de proteínas.

Heurística
É um processo de pensamento analítico para resolução de problemas complexos que não segue um procedimento padronizado de uma série de passos ("algoritmo"), mas utiliza análise da experiência prévia de problemas semelhantes, análise de casos similares, aplicação de pensamento ou enfoque sistêmico e outras técnicas de pensamento criativo para formar uma tentativa de solução em forma de uma hipótese a ser testada na prática.

Introns São regiões não codificadoras (interrompem a síntese protéica) dentro de um gene. Eles são transcritos em *RNA*, mas são removidos por *splicing* antes da síntese de proteínas.

Marcadores do tipo Microssatélites
Também conhecidos como STRs, estes marcadores constituem uma poderosa classe de marcadores moleculares em análise genômica.

Marcadores Moleculares
São moléculas como *DNA* ou proteínas que marcam uma região(ões) do genoma ligada(s) a alguma característica genética que está sendo estudada.

Metódo *neighbor-joining*
É um método utilizado para construção de árvores filogenéticas. Uma medida da distância entre um conjunto de seqüências deve ser determinado. Baseado nesta distância, um algoritmo com o método *neighbor-joining* encontrará os dois conjuntos mais próximos e irá agrupá-los. Este passo se repete até que todas as seqüências sejam colocadas dentro da árvore.

Motif (Motivos)

É um elemento conservado de um alinhamento de seqüência que normalmente correlaciona-se com uma função particular. Motivos são gerados de um alinhamento múltiplo, correspondentes a uma região cuja função ou estrutura é conhecida. Motivos predizem regiões estruturais ou funcionais em qualquer outra seqüência pertencente a família original (seqüências homólogas). Uma base de dados comumente utilizada para análise de motivos é a base *Pfam*.

Northern Blot

É uma técnica usada na pesquisa em Biologia Molecular para estudar a expressão gênica, ou seja, verificar e quantificar se um determinado gene de um genoma é ou não transcrito em *RNA*.

Open Reading Frame (ORF)

ORF é uma porção do cromossomo de um organismo que contém uma seqüência de bases (genes) que podem potencialmente codificar uma proteína. A existência de uma *ORF*, especialmente uma longa, é uma boa indicação da presença de um gene.

Organismos eucariotos

São organismos que possuem células eucariotas (do grego *eu*, bom, e *karya*, núcleo), as quais indicam a existência de um núcleo bem definido, perfeitamente demarcado pela membrana celular. Exemplos de organismos eucariotos são plantas e animais.

Organismos procariotos

São os organismos que possuem células procariontes (do grego *pro*, primitivo, e *karya*, núcleo), que não possuem uma membrana celular que separe o nucleóide, uma estrutura correspondente ao núcleo, do líquido citoplasmático. Exemplos de organismos procariotos são as bactérias.

Ortólogos

São genes homólogos que compartilham funções e descendem de um ancestral comum que, no processo evolutivo, divergiu para espécies diferentes, ou seja, consistem de genes homólogos presentes em organismos diferentes.

Parálogos São genes homólogos gerados por eventos de duplicação, que ocupam duas diferentes posições no mesmo genoma.

Phred É um *software* que analisa uma seqüência "bruta"(original) para produzir uma seqüência de base com uma pontuação de qualidade (*quality score*) associada para cada posição na seqüência. Análise da qualidade de cromatogramas.

Produto de Gene

É o material bioquímico, *RNA* ou proteína, resultante da expressão de um gene. A quantidade de produto de gene é utilizada para medir

o quão um gene é ativo. Quantidades anormais podem ser correlacionadas com doenças causados por alelos (variações menores de um mesmo gene).

Profiles *Profiles*, ou perfis de alinhamento, são uma representações numéricas, normalmente matricial, de um alinhamento múltiplo. Esse perfil representa as características comuns àquele particular conjunto de seqüências, que é, em geral, uma família de proteínas. Uma outra definição é de que perfis são tabelas (matrizes) que listam as freqüências de cada aminoácido em cada posição de uma seqüência protéica. As freqüências são calculadas de um alinhamento múltiplo de seqüências contido em um dado domínio de interesse. A análise por perfis usa o fato que certas posições em uma família de seqüências são mais conservadas que outras posições.

Proteínas de superfície
São proteínas encontradas na superfíce (membrana) de uma célula.

Proteínas São formadas por cadeias de aminoácidos, localizadas nos genes que codificam as codificam. As proteínas são essencias a estrutura, funcionamento e ao regulamento do organismo dos seres vivos. São exemplos de proteínas: hormônios, enzimas e anticorpos.

Proteoma
É o conjunto de proteínas expressas por um genoma.

RNA Ácido nucléico envolvido na transferência da informação genética e sua decodificação em uma cadeia polipeptídica. Em alguns vírus, o *RNA* é o material genético primário. Localiza-se no núcleo das células e no citoplasma, participando na síntese de *DNA* quando as células se multiplicam. O *RNA* se diferencia do *DNA* em três aspectos: os nucleotídeos do *RNA* contém o açúcar ribose ao invés do desoxirribose; o *RNA* contém a base uracila ao invés da timina; e o *RNA* é uma molécula de fita simples ao invés de uma hélice de dupla fita.

rRNA Moléculas de *RNA* que transportam os aminoácidos durante a síntese protéica.

Singlets São seqüências que não foram agrupadas com nenhuma outra (por não possuir similaridade com nenhuma outra seqüência). *Singlets* são formados por uma única seqüência.

Splicing (Processamento)
Trata-se do processo de remoção de íntrons do *mRNA*.

Taxonomia
Refere-se a classificação das coisas e aos princípios subjacentes da classificação. Quase tudo - objetos animados, inanimados, lugares e eventos - pode ser classificado de acordo com algum esquema taxonômico.