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

Implementação de Protótipo de um SMA para Anotação Manual em Projetos de Seqüenciamento de Genomas

Hugo Wruck Schneider
Anderson Gray Frazzon Pereira

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientadora
Profª. Drª. Maria Emilia Telles Walter

Coorientadora
Profª. Drª. Célia Ghedini Ralha

Brasília
2006

Universidade de Brasília – UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Curso de Bacharelado em Ciência da Computação

Coordenadora: Profª. Drª. Cláudia Nalon

Banca examinadora composta por:

Profª. Drª. Maria Emilia Telles Walter (Orientadora) – CIC/UnB
Profª. Drª. Célia Ghedini Ralha (Coorientadora) – CIC/UnB
Prof. Dr. Roberto Togawa – Embrapa/Recursos Geneticos e Biotecnologia

CIP – Catalogação Internacional na Publicação

Schneider, Hugo Wruck.

Implementação de Protótipo de um SMA para Anotação Manual em Projetos de Seqüenciamento de Genomas / Hugo Wruck Schneider, Anderson Gray Frazzon Pereira. Brasília : UnB, 2006.
97 p. : il. ; 29,5 cm.

Monografia (Graduação) – Universidade de Brasília, Brasília, 2006.

1. anotação manual, 2. sistema multiagentes,
3. bioinformática, 4. projetos de sequenciamento de genomas

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

Implementação de Protótipo de um SMA para Anotação Manual em Projetos de Seqüenciamento de Genomas

Hugo Wruck Schneider
Anderson Gray Frazzon Pereira

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Profª. Drª. Maria Emilia Telles Walter (Orientadora)
CIC/UnB

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

Prof. Dr. Roberto Togawa
Embrapa/Recursos Geneticos e Biotecnologia

Profª. Drª. Cláudia Nalon
Coordenadora do Bacharelado em Ciência da Computação

Brasília, 07 de dezembro de 2006

Dedicatória

Dedicamos esse trabalho a nossas famílias.

Hugo Wruck Scnheider e Anderson Gray Frazzon Pereira

Agradecimentos

Agradeço a minha família por todo apoio e toda ajuda nessa grande jornada e em toda minha vida e por todas as oportunidades que, por eles, me foram dadas. Agradeço a meu grande amigo Anderson que compartilhou todo esse trabalho duro, o qual não teria sido concluído sem nosso trabalho em grupo, e agradeço também por sua amizade e companheirismo em todos outros momentos. Agradeço a Professora Maria Emília e a Professora Célia que acreditaram em nossa capacidade de tornar esse projeto possível e que nos ajudaram a alcançar nossos objetivos. Também agradeço a todos os professores que contribuiram em minha formação. Agradeço a Cynthia por sempre ter estado ao meu lado nos momentos felizes e também nos momentos difíceis. Agradeço também a todos meus amigos que estiveram junto de mim em todos os momentos da minha vida.

Hugo Wruck Scnheider

Primeiramente agradeço a minha família que, com muita luta, me proporcionaram oportunidades para ingressar nesta universidade. Agradeço às minhas orientadoras, Maria Emília e Célia, por sua atenção e dedicação com as quais acompanharam este trabalho. Agradeço ao meu amigo Hugo e todo o seu empenho e as inúmeras horas de trabalho neste projeto, as quais tornaram esse projeto possível. Agradeço aos meu amigos por estarem sempre presentes nas horas difíceis. Agradeço à minha amiga Ingrid por ter me acompanhado neste caminho.

Anderson Gray Frazzon Pereira

Por fim, agradecemos a Deus

Sumário

Lista de Figuras **10**

Lista de Tabelas **13**

Capítulo 1 Introdução **14**

Capítulo 2 Fundamentos de Biologia Molecular **19**

2.1 Proteínas 19
2.2 Ácidos Nucléicos 22
2.2.1 DNA 22
2.2.2 RNA 23
2.3 Genes, Cromossomos e Síntese Protéica 24
2.3.1 Síntese Protéica 26
2.3.2 Fases de Leitura 31
2.3.3 RNA Não-Codificador (ncRNA) 31
2.4 Estudando o Genoma 32
2.4.1 Seqüenciamento 33
2.4.2 Genoma Estrutural e Funcional 35

Capítulo 3 Conceitos Básicos de Bioinformática **38**

3.1 *Pipeline* de um projeto de seqüenciamento de genoma 39
3.2 Anotação Genômica 42
3.2.1 Ferramentas de Análise 43

Capítulo 4 Inteligência Artificial **47**

4.1 Representação do Conhecimento 47
4.1.1 Regras de Inferência 49
4.2 Sistema Multiagente 50
4.2.1 Agentes Inteligentes 50
4.2.2 Sistema Multiagente 52

4.2.3 FIPA 53
4.2.4 Ontologia 56
4.2.5 Ontologia aplicada à Bioinformática 57
4.3 Ferramentas para SMA 57
4.3.1 JADE 58

Capítulo 5 Um SMA para Anotação Manual 62
5.1 Arquitetura 62
5.2 Funcionalidades da Plataforma 65

Capítulo 6 Aspectos da Implementacionais e Discussão dos Resultados Obtidos 66
6.1 Ontologias 66
6.2 *Parsers* 66
6.3 Bases de Conhecimentos 67
6.4 Comunicação entre os agentes 68
6.5 Agentes 69
6.6 Discussão 71

Capítulo 7 Conclusões e Trabalhos Futuros 73

Apêndice A Código JESS 75

Apêndice B ENIA 2007 77

Apêndice C IWGD 2007 90

Resumo

Atualmente, existe um grande volume de dados contendo informações biológicas que estão disponíveis na *Web*. Os biólogos usam tais informações para inferir funções de genes que são descobertos e estudados em projetos de seqüenciamento de genomas. Esta tarefa constitui a fase de anotação destes projetos. Ferramentas computacionais podem auxiliar os biólogos a completarem esta tarefa com uma maior acurácia e em menor tempo. O objetivo do nosso trabalho é disponibilizar aos biólogos sugestões de funções de genes, que podem apoiar a anotação manual em projetos genômicos usando técnicas de Inteligência Artificial. Particularmente, este trabalho propõe um sistema colaborativo usando a abordagem de Sistemas Multiagente, baseado em fonte de dados genômicos integrados, heterogêneos e autonômos disponíveis na Web. O sistema provê a combinação de diferentes agentes com ontologias específicas. Estes agentes devem interagir e fornecer como resultado uma sugestão de anotação que deve ser posteriormente avaliada e validada pelos biólogos. Utilizando este sistema apresentamos um estudo de caso comparando os resultados produzidos pelo nosso Sistema com as anotações manuais já completadas no Projeto Genoma *Paracoccidioides brasiliensis*, realizado pela Rede Genoma Centro-Oeste.

Palavras-chave: anotação manual, sistema multiagentes, bioinformática, projetos de sequenciamento de genomas

Abstract

Nowadays there is a great volume of data containing biological information available on the web. The biologist use this information to infer functions for genes discovered and studied in genome sequencing projects. This task is done on the annotation phase inside there projects. Computational tools can help biologist to complete this task with more accuracy and less time. The objective of this work is to sugest an annotation on a genomic project to the biologist using artificial intelligence techniques. Particularly, this work propose a collaborative system using a multiagent system approach based on integrated, heterogeneous and autonomous genome data source. Our system provides the combination from distinct agents having specific ontologies. These agents must interact and produce as a result an annotation suggestion, that must be later validated by the biologists. Using this system, we present a case study comparing annotations already made for the Paracoccidioides brasiliensis genome project, developed by the Midwest Genome Network.

Keywords: annotation, multiagent system, bioinformatics ,genome sequencing projects

Lista de Figuras

2.1 Estrutura molecular dos aminoácidos. 20
2.2 Uma proteína é formada pela ligação do grupo amina (NH_2) de um aminoácido com o grupo carboxila (COOH) de outro aminoácido, formando uma amida (em destaque). 21
2.3 Moléculas de desoxirribose e de ribose. 22
2.4 Esta figura mostra o esquema das ligações e a estrutura espacial do DNA. (A) O radical fosfato liga o carbono $3'$ de um nucleotídeo com o carbono $5'$ do próximo nucleotídeo. O conjunto dessas ligações forma o chamado esqueleto fosfato-açúcar. (B) O pareamento das bases liga as duas fitas de DNA. (C) As duas fitas formam espacialmente uma dupla hélice. 23
2.5 Desenho esquemático de uma molécula de RNA (tig). 24
2.6 Visão esquemática dos genes dentro de um cromossomo. O conjunto dos cromossomos numa célula é chamado de genoma. 26
2.7 Visão esquemática dos íntrons e éxons dentro do gene. 26
2.8 Excisão alternativa do gene α-tropomiosina de ratos (regula a contração nas células musculares). 27
2.9 Visão esquemática da estrutura molecular de um tRNA. Observe a presença de bases não-convencionais (ψ). 28
2.10 Visão esquemática do processo de síntese protéica. Etapa 1: A lisina é ligada à cadeia protéica, enquanto outro tRNA pareia com o próximo códon. Etapa 2: A lisina é desligada do tRNA, enquanto o triptofano é ligado à lisina. Etapa 3: O tRNA da lisina sai do ribossomo. Etapa 4: O triptofano é desligado do tRNA. Etapa 5: A proteína é completada e solta no citoplasma e a maquinaria de síntese (ribossomo + mRNA + tRNA) é desfeita. 29

2.11 (A) Desenho esquemático mostrando vários ribossomos traduzindo uma mesma molécula de mRNA. O conjunto formado por esses ribossomos mais a molécula de mRNA é denominado polirribossomo. (B) Microfotografia eletrônica de um polirribossomo de uma célula eucariótica. 30
2.12 Dogma Central da Biologia Molecular (33). 30
2.13 Resultado da eletroforese em gel em filme fotográfico. 34
2.14 Exemplo de eletroferograma gerado por um seqüenciador automático. 35
2.15 Ciclo de produção deESTs. 36

3.1 As três grande fases de um *Pipeline* para um projeto de seqüenciamento de genoma. 39
3.2 Exemplo de *Pipeline* da etapa de Submissão do Projeto Pb. . . 41
3.3 Exemplo de um arquivo no formato fasta. A característica principal é que a primeira linha possui o caractere especial > seguido por um texto e as próximas linhas contem a seqüência. 41
3.4 Exemplo de *pipeline* da etapa de Montagem do Projeto Genoma Pb. 42
3.5 Exemplo de *pipeline* da etapa de Anotação do Projeto Genoma Pb. 43

4.1 Características de um agente inteligente (30). 51
4.2 Estrutura de um agente na abordagem de um Sistema Multiagente (iwg). 53
4.3 Modelo de referência para gerenciamento de agentes segundo a FIPA (36) 54
4.4 Arquitetura do *framework* JADE (9). 59

5.1 Proposta da arquitetura do SMA para anotação manual (iwg) 63

6.1 Parte da implementação da ontologia do BLAST no Protégé . 67
6.2 Diagrama do protocolo de comunicação Responder (9) 69

B.1 A arquitetura em três camadas do sistema *BioAgents*. 82
B.2 *Screenshot* da tela de execução e do *sniffer* dos agentes do *BioAgents* no framework *JADE*. 84
B.3 Conjunto de regras *Jess* para análise de saídas *BLAST* e *FASTA*. 85

C.1 BioAgents Architecture 91

Lista de Tabelas

2.1 Representação do DNA por uma seqüência de letras, onde cada letra representa uma base. A orientação de cada fita é indicada por $5'$ e $3'$, sendo as duas fitas orientadas de forma contrária. 23
2.2 O Código Genético para RNA 25
2.3 Um seqüência de DNA com as seis possíveis fases de leitura . 31
2.4 Cada coluna indica todos os fragmentos da seqüência original terminados em uma base particular. 33

6.1 Tabela de resultados obtidos pelo SMA comparados com o Projeto Genoma Pb 72

B.1 Resultados do *BioAgents* utilizando dados do Projeto Genoma Pb. 86

Capítulo 1

Introdução

Desde que a estrutura de dupla hélice de uma molécula de DNA foi descoberta por Watson e Crick em 1953 (39), pesquisadores de todo o mundo têm despendido grandes esforços para melhor compreender a estrutura e o funcionamento da genética dos seres vivos. Desde o início dos anos 90, os avanços nos métodos e nas técnicas relacionadas à Biologia Molecular e a Bioinformática possibilitaram acelerar muito o processo de descoberta e descrição da estrutura e das funcionalidades dos genes.

O Projeto Genoma Humano, iniciado em 1990 e finalizado em 2001 (38; 21), foi de essencial importância para o atual desenvolvimento da Biologia Molecular e da Bioinformática no mundo. Muitos países tiveram a oportunidade de participar desse projeto. Paralelamente a esse grande projeto foram desenvolvidos outros projetos menores de seqüenciamento de organismos mais simples, como *Saccharomyces cerevisiae*, *Caenorhabditis elegans*, *Drosophilamelanogaster* e *Arabidopsis thaliana*, entre outros. Estes projetos visavam a geração de dados e o aperfeiçoamento das técnicas de análise de seqüências biológicas para que depois fossem usados no seqüenciamento do genoma humano. Após um grande investimento de recursos e pesquisa, estes esforços proporcionaram o surgimento de uma nova era, que pode ser chamada de Era Genômica da Biologia (18).

Estes projetos geraram um grande volume de seqüências biológicas e informações sobre elas, assim como criaram novas metodologias de pesquisa. Surgiram máquinas de seqüenciamento de alta precisão, com custos relativamente baixos, além de inúmeras ferramentas computacionais de análise e automação do processo como um todo. Esses novos recursos têm possibilitado aos pesquisadores explorarem cada vez mais rapidamente os padrões de expressão gênica dos genomas, tanto considerando genes quanto o DNA

no núcleo dos organismos. A análise e anotação dos genes pode ser realizada de forma interativa através da *Web* usando uma interface gráfica amigável.

Um grande volume de recursos financeiros e humanos vem sendo investido em Bioinformática, como se pode notar pelo grande número de centros privados e governamentais especializados em seqüenciamento genômico. Um exemplo é o *The Institute for Genome Research - TIGR -* (tig), um dos principais centros de pesquisa de geração de dados de seqüências genômicas. Vários outros centros podem ser citados, como o *Sanger Centre* (san), o *DOE Joint Genome Institute* (doe), a Universidade de Washington em St. Louis (wus), entre outros. Uma lista completa dos projetos de seqüenciamento de genomas e dos centros de pesquisa pode ser acessada no site do *Genomes OnLine Database - GOLD -* (gol).

Em âmbito nacional, os primeiros esforços vieram da Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP - e do CNPq, que foram as primeiras instituições a apoiar projetos na área de seqüenciamento de genomas no Brasil. Inicialmente, a FAPESP e o CNPq formaram um consórcio de laboratórios e criaram um instituto virtual responsável pelo seqüenciamento e análise de nucleotídeos, denominado *Organization for Nucleotide Sequencing and Analysis* - ONSA - (ons). O primeiro resultado importante desse instituto e o seu reconhecimento internacional ocorreram com a publicação do genoma do fitopatógeno *Xylella fastidiosa*, agente etiológico da *Citrus Variegated Chlorosis* (CVC), mais conhecida como praga do amarelinho. Essa doença destrói lavouras de laranja, principalmente no Estado de São Paulo, ocasionando prejuízos econômicos de grandes proporções. Tal feito mereceu uma publicação na revista Nature (34).

O sucesso desse projeto foi tão significativo que motivou os orgãos governamentais, particularmente a FAPESP, a investir em outros projetos, como o do mapeamento do genoma da cana-de-açúcar, do câncer humano, que teve a participação do Instituto Ludwig para Pesquisa do Câncer, do café e também de vários organismos e pragas como o *Xylella fastidiosa* de videira, o *Xanthomonas campestris*, o *Xanthomonas axonopodis* e o *Leifsonia xyli*, além de subsidiar outros projetos como o do mapeamento do genoma funcional do *Schistosoma mansoni*.

Outro ponto que merece destaque consistiu na criação das redes de seqüenciamento, estimulado pelo governo federal, que ocorreu tanto no âm-

bito nacional como no regional. No âmbito nacional foram criados dois projetos: o Projeto Genoma Brasileiro, que seqüenciou a bactéria *Chromobacterium violaceum* (37), que apresenta resultados de potencial aplicabilidade no controle da doença de chagas e da leishmaniose, e o projeto Genolyptus (1), responsável pelo seqüenciamento do eucalipto (Fundo Verde-Amarelo/MCT). No âmbito regional surgiram várias redes com projetos de seqüenciamento de organismos importantes, especialmente para o controle de pragas e doenças. Dentre elas podemos destacar:

- Rede Genoma do Estado de Minas Gerais (*Schistosoma mansoni*);
- Rede Genoma Nordeste (*Leishmania chagasi*);
- Rede Genômica do Estado da Bahia e São Paulo (*Crinipellis perniciosa*);
- Rede Genoma integrante do Consórcio do Instituto de Biologia Molecular do Paraná, FIOCRUZ e Universidade de Mogi das Cruzes (*Trypanossoma cruzi*);
- Programa Genoma do Estado do Paraná (*Herbaspirillum seropedicae*);
- Rede Genoma do Rio de Janeiro (*Gluconacetobacter diazotrophicus*);
- Rede Sul de Análise de Genomas e Biologia Estrutural (*Mycoplasma hyopneumoniae*) e
- Rede Genoma Centro-Oeste (*Paracoccidioides brasiliensis*).

Estes projetos colocam o Brasil no grupo dos países com tecnologia e infra-estrutura suficientes para viabilizar e conduzir pesquisas na área genômica. Isso é de vital importância estratégica, pois permite que o país desenvolva tecnologia própria para resolver problemas específicos que afetam nossa população e/ou produção agropecuária, independente da boa vontade dos governos e laboratórios estrangeiros. Além disso, esses projetos estimulam o desenvolvimento de tecnologias e a capacitação de profissionais especializados, o que contribui para colocar o Brasil em posição de igualdade perante a comunidade científica internacional nessa área.

Os projetos de seqüenciamento, de modo geral, são implementados usando um conjunto de programas e procedimentos denominado de *pipeline*.

Um *pipeline* é composto por três fases, executadas em seqüência, denominadas:

- Submissão - é a primeira fase, onde os biólogos enviam para o laboratório de Bioinformática os fragmentos de seqüências, geradas por um sequenciador automático. Estes fragmentos serão convertidos em um formato adequado para as fases seguintes.

- Montagem - nessa fase os genes que possivelmente vieram da mesma região do DNA são agrupados. Um grupo é formado por seqüências que têm similaridade e são identificadas por uma única seqüência, chamada de seqüência consenso.

- Anotação Automática/Manual - nesta fase, as seqüências consenso são comparadas com seqüências de bancos de dados visando associar funções a estas seqüências. Na fase automática são feitas comparações utilizando programas e associadas funções aos genes como resultado destas comparações. Na fase manual o biólogo deve usar seus conhecimentos e experiência para decidir qual é realmente a função do gene analisado.

Um Agente Inteligente, em Inteligência Artificial (IA), pode ser definido como uma entidade de *software* capaz de perceber o ambiente onde se encontra por meio de sensores tendo capacidade de interagir com tal ambiente por meio de atuadores. Um Sistema Multiagente (SMA) consiste em um grupo de agentes inteligentes que interagem entre si, por meio de troca de mensagens através de uma infra-estrutura de rede computacional ou por um *software* que viabilize tal comunicação.

Neste contexto, os objetivos deste trabalho são:

- Desenvolver, utilizando técnicas de SMA, um sistema de apoio à fase de anotação manual para projetos de seqüenciamento de genomas. Este sistema poderá agregar uma maior qualidade à anotação manual e os biólogos poderão diminuir o tempo para concluírem suas anotações;

- Realizar o estudo de caso utilizando as anotações realizadas no Projeto Genoma Pb da Rede Genoma Centro-Oeste (pb), discutindo e comparando os resultados obtidos pelo SMA com as anotações manuais já completadas pelos biólogos.

O restante deste documento está distribuído da seguinte forma.

No Capítulo 2 são apresentados fundamentos de Biologia Molecular. No Capítulo 3 são apresentados os principais conceitos e ferramentas da Bioinformática que são utilizados em projetos de seqüenciamento. No Capítulo 4 são expostos os fundamentos de SMA, bem como as principais ferramentas de SMA que foram utilizadas neste trabalho. No Capítulo 5 é apresentado a arquitetura definida para o SMA proposto, bem com os aspectos implementacionais do protótipo desenvolvido. No Capítulo 6.6, são apresentados os resultados obtidos no estudo de caso. Por fim, no Capítulo 7, são conclusões e apresentados as feitas sugestões de trabalhos futuros.

Capítulo 2

Fundamentos de Biologia Molecular

Neste capítulo serão apresentados conceitos básicos em Biologia Molecular que serão utilizados neste trabalho.

Na Seção 2.1 explicaremos a estrutura das proteínas. Na Seção 2.2 apresentaremos os dois tipos de ácidos nucléicos presentes em organismos vivos: o DNA e o RNA. Na Seção 2.3 definiremos genes e cromossomos além de apresentarmos uma breve descrição do mecanismo de síntese de proteínas que ocorre no interior das células, além de e conceitos de RNA não codificador. Finalmente, na Seção 2.4, serão apresentados algumas técnicas biológicas utilizadas no estudo de genomas.

2.1 Proteínas

Proteínas são construídas por um encadeamento de moléculas mais simples denominadas aminoácidos. Toda a imensa variedade de proteínas existentes é formada por apenas 20 aminoácidos, que se repetem numa seqüência característica para cada proteína (10). Os aminoácidos são formados por um radical amina (NH_2) e um radical carboxila (COOH). Além destes, compõem o aminoácido um átomo de carbono central C_α, um átomo de hidrogênio e uma cadeia lateral, que é o único elemento que varia para cada aminoácido. A estrutura molecular de cada um desses aminoácidos pode ser observada na Figura 2.1.

As proteínas fazem parte do grupo dos polipeptídeos, pois os aminoácidos que as formam são unidos por ligações peptídicas. Esta ligação representa a união do grupo amina (NH_2) de um aminoácido com o grupo

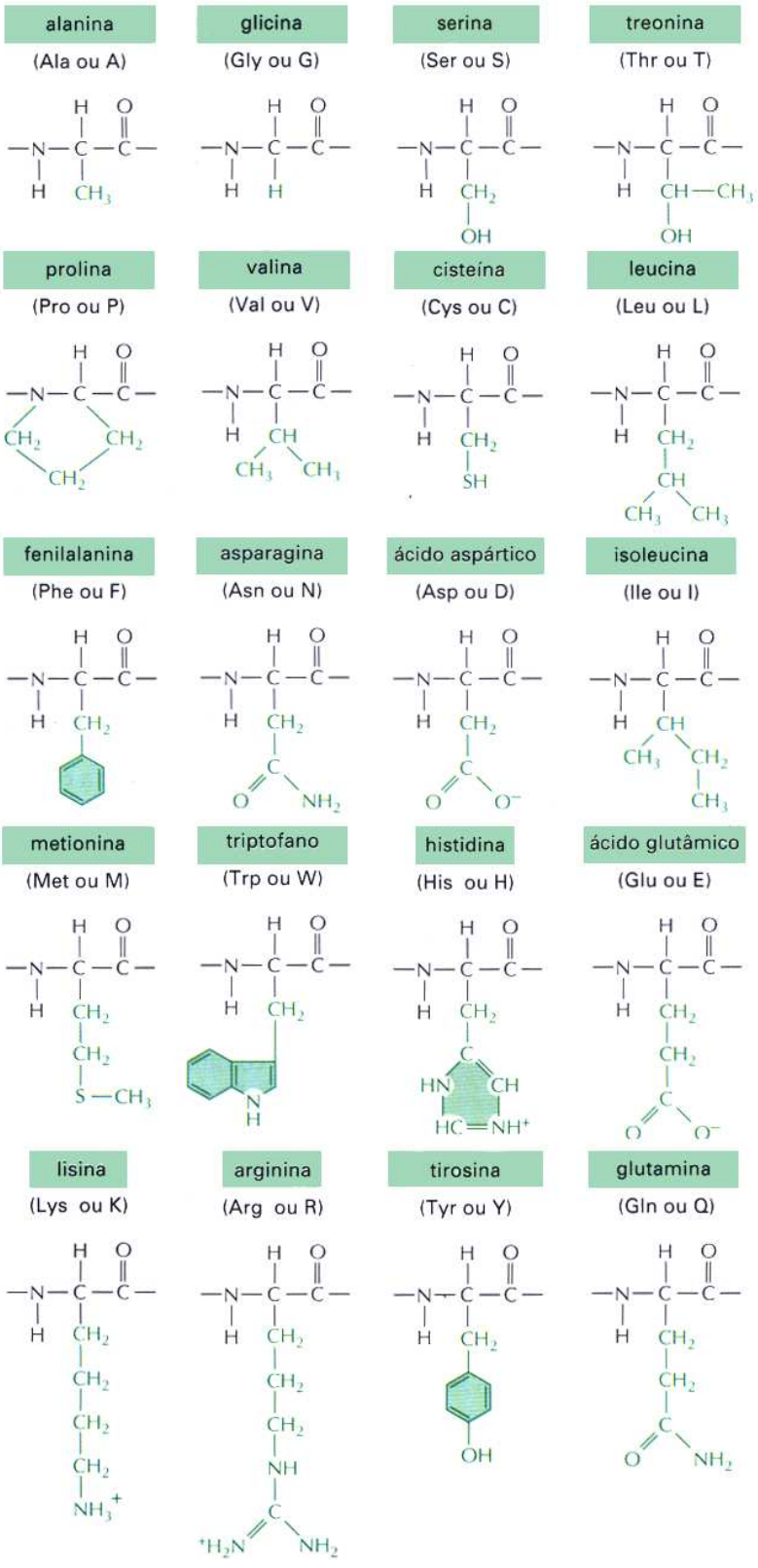

Figura 2.1: Estrutura molecular dos aminoácidos.

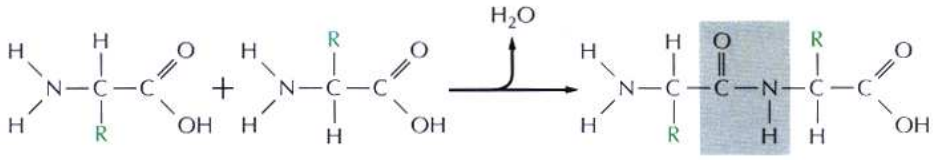

Figura 2.2: Uma proteína é formada pela ligação do grupo amina (NH_2) de um aminoácido com o grupo carboxila (COOH) de outro aminoácido, formando uma amida (em destaque).

carboxila (COOH) de outro aminoácido, através da formação de uma amida e da liberação de uma molécula de água (Figura 2.2). Quando os aminoácidos se ligam numa cadeia protéica nessas condições, são chamados de **resíduos**. As ligações peptídicas permitem que sejam formadas longas cadeias de aminoácidos. A seqüência dos aminoácidos que constituem as proteínas, conhecida como **estrutura primária**, determina a forma e a função da proteína. As interações moleculares entre os aminoácidos mais próximos fazem com que a cadeia protéica assuma ainda uma **estrutura secundária** e, as interações entre regiões mais distantes dentro das proteínas constituem a **estrutura terciária**. As duas últimas referem-se à disposição espacial da molécula, enquanto a estrutura primária diz respeito somente à seqüência de aminoácidos (10).

É possível também classificar proteínas com base em sua função. Elas podem ser divididas em dois grupos: proteínas estruturais e proteínas biologicamente ativas. Algumas proteínas, entretanto, podem pertencer aos dois grupos. As proteínas estruturais são compostas por cadeias alongadas e têm como função compor estruturas de órgãos, tecidos, etc. Dois bons exemplos nos animais são o colágeno (ossos, tendões, pele e ligamentos) e a queratina (unhas, cabelos, penas e bicos). A maioria das proteínas biologicamente ativas têm forma globular, sendo responsáveis pela execução e controle da maior parte das atividades dentro do organismo. Exemplos são as enzimas (indispensáveis para viabilizar grande parte das reações químicas), os hormônios protéicos (que atuam como mensageiros químicos), as proteínas de transporte (como as lipoproteínas, que podem carregar o colesterol) e as imunoglobulinas (ou anticorpos), que protegem o corpo de microorganismos invasores (10).

2.2 Ácidos Nucléicos

Os ácidos nucléicos estão presentes em todos os seres vivos e contêm as informações necessárias para a síntese de todas as proteínas que cada organismo é capaz de produzir. Existem dois tipos de ácidos nucléicos em organismos vivos, o **RNA** - ácido ribonucléico e o **DNA** - ácido desoxirribonucléico.

2.2.1 DNA

O **DNA** é uma molécula formada por duas seqüências (ou **fitas**) de moléculas mais simples chamadas nucleotídeos. Os nucleotídeos constituem a unidade básica do DNA e são compostos por uma molécula de açúcar (no caso, a desoxirribose), um fosfato e uma base nitrogenada. A desoxirribose é composta por 5 átomos de carbono numerados de $1'$ a $5'$ (Figura 2.3).

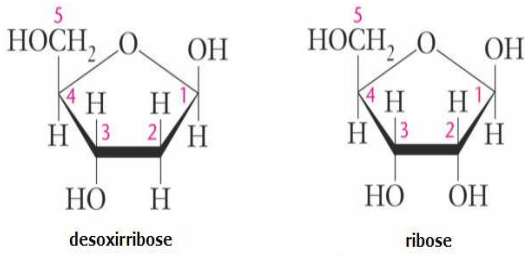

Figura 2.3: Moléculas de desoxirribose e de ribose.

Existem quatro tipos de bases nitrogenadas presentes no DNA: adenina (A), guanina (G), citosina (C) e timina (T). As duas fitas do DNA formam uma estrutura helicoidal (dupla hélice) e encontram-se ligadas por meio de pontes de hidrogênio que se estabelecem entre as bases A e T e entre as bases C e G (Figura 2.4). Por isso, dizemos que as bases A e T são complementares entre si, o mesmo ocorrendo com as bases C e G. Os pares de bases complementares (pb) fornecem uma unidade de comprimento para o DNA.

Devido a estrutura química das ligações das bases, cada uma das fitas possui uma **orientação**, que é indicada denominando-se as extremidades de cada fita por $5'$ e $3'$, respectivamente (33). Esses números vêm da numeração dos carbonos da desoxirribose (Figura 2.3). Denota-se o DNA por uma seqüência de letras, onde cada letra representa uma base. A fita du-

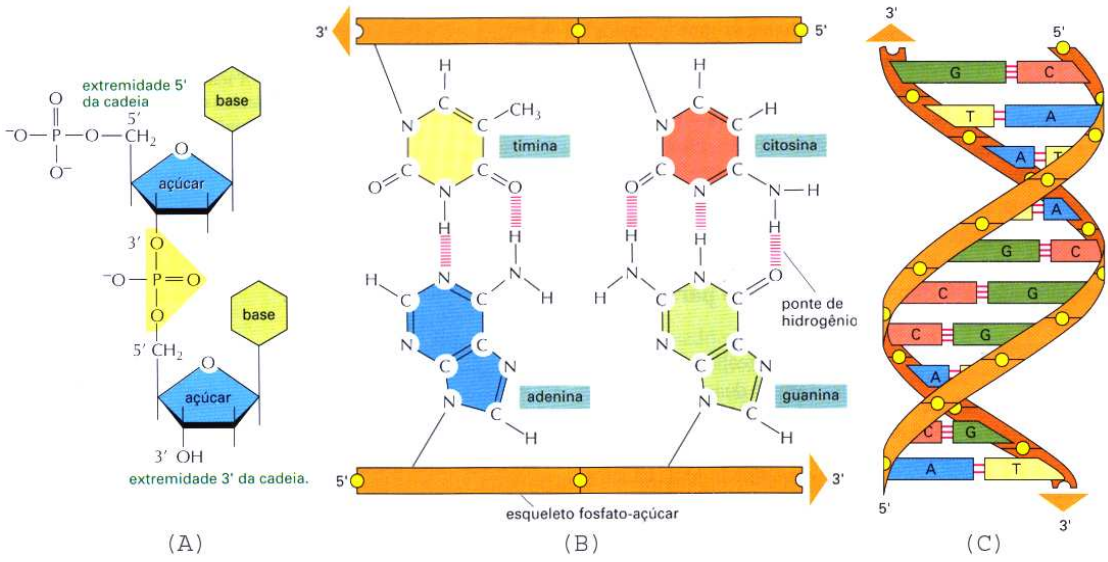

Figura 2.4: Esta figura mostra o esquema das ligações e a estrutura espacial do DNA. (A) O radical fosfato liga o carbono $3'$ de um nucleotídeo com o carbono $5'$ do próximo nucleotídeo. O conjunto dessas ligações forma o chamado esqueleto fosfato-açúcar. (B) O pareamento das bases liga as duas fitas de DNA. (C) As duas fitas formam espacialmente uma dupla hélice.

pla é representada colocando-se uma fita sobreposta à outra. As duas fitas são orientadas de forma contrária, de modo que a extremidade $5'$ de uma corresponde à extremidade $3'$ da outra (Figura 2.1).

$$5' \ldots \text{TACTGAA} \ldots 3'$$
$$3' \ldots \text{ATGACTT} \ldots 5'$$

Tabela 2.1: Representação do DNA por uma seqüência de letras, onde cada letra representa uma base. A orientação de cada fita é indicada por $5'$ e $3'$, sendo as duas fitas orientadas de forma contrária.

A importância das moléculas de DNA reside no fato de nelas estarem codificadas todas as informações necessárias para construir cada proteína ou RNA encontrado no organismo, garantindo assim a sobrevivência do indivíduo e a perpetuação da espécie.

2.2.2 RNA

As moléculas de RNA são similares às moléculas de DNA, sendo também compostas de nucleotídeos. No entanto, elas apresentam algumas difer-

enças importantes:

- em geral, apresentam uma única fita de nucleotídeos, que pode assumir inúmeras conformações espaciais (Figura 2.5), a exemplo das proteínas;
- o açúcar é a ribose e não a desoxirribose (Figura 2.3);
- não há a ocorrência da base timina, em seu lugar está presente a **uracila**. A uracila forma pontes de hidrogênio com a adenina de forma semelhante à timina;
- enquanto o DNA tem essencialmente uma única função (codificar informações), há diferentes tipos de RNA, cumprindo as mais diversas funções dentro da célula.

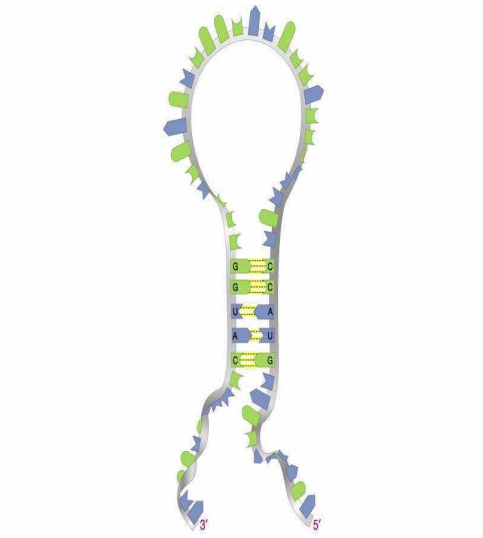

Figura 2.5: Desenho esquemático de uma molécula de RNA (tig).

2.3 Genes, Cromossomos e Síntese Protéica

Genes são trechos da molécula de DNA que contêm as informações que determinam as características de uma espécie como um todo e dos indivíduos em si (4). A cada proteína produzida dentro de um organismo corresponde, em geral, um gene na molécula de DNA. A informação armazenada nesses genes representa a codificação da seqüência de aminoácidos de cada proteína.

Segundo Lewin (23), em 1961 foi descoberto que essa codificação é feita através de triplas de nucleotídeos, chamadas de **códons**, onde cada tripla corresponde a um aminoácido em uma proteína. A cada códon corresponde também um **anticódon**, que é a tripla contendo as bases complementares às bases do códon original. Existem 64 códons diferentes, considerando todas as possíveis combinações de três bases. Como há apenas 20 aminoácidos a serem codificados, alguns deles são codificados por mais de um códon. A correspondência entre cada códon e cada aminoácido é dada pelo código genético. A Tabela 2.2 apresenta o código genético para RNA utilizado pela maioria dos organismos. Além de códons que mapeiam aminoácidos, há também códons para indicar o final de uma proteína, os chamados **códons de terminação** (*STOP codons*) e códons para marcar o início da proteína (*START codons*) ou códon de início.

Primeira Posição	*Segunda Posição*				*Terceira Posição*
	G	A	C	U	
G	Gly	Glu	Ala	Val	G
	Gly	Glu	Ala	Val	A
	Gly	Asp	Ala	Val	C
	Gly	Asp	Ala	Val	U
A	Arg	Lys	Thr	Met	G
	Arg	Lys	Thr	Ile	A
	Ser	Asn	Thr	Ile	C
	Ser	Asn	Thr	Ile	U
C	Arg	Gln	Pro	Leu	G
	Arg	Gln	Pro	Leu	A
	Arg	His	Pro	Leu	C
	Arg	His	Pro	Leu	U
U	Trp	STOP	Ser	Leu	G
	STOP	STOP	Ser	Leu	A
	Cys	Tyr	Ser	Phe	C
	Cys	Tyr	Ser	Phe	U

Tabela 2.2: O Código Genético para RNA

Um **cromossomo** é uma molécula muito longa de DNA que contém muitos genes. O conjunto completo dos cromossomos de uma célula é denominado **genoma**. O número de cromossomos no interior de cada célula é característico de cada espécie. As células humanas, por exemplo, possuem 46 cromossomos, enquanto as de um camundongo possuem 40 cromossomos.

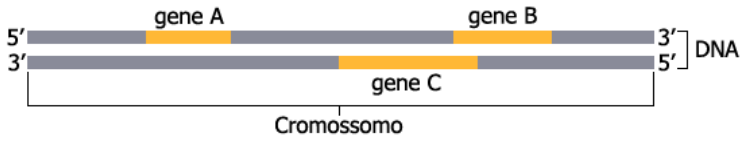

Figura 2.6: Visão esquemática dos genes dentro de um cromossomo. O conjunto dos cromossomos numa célula é chamado de genoma.

2.3.1 Síntese Protéica

Mecanismos celulares reconhecem um gene através de um **promotor**, uma região no DNA que precede cada gene, indicando seu início.

Tendo sido reconhecida a posição inicial do gene no DNA, uma cópia dos nucleotídeos que o compõem é feita numa molécula de RNA, através do processo conhecido por **transcrição**. Essa molécula de RNA é denominada **RNA mensageiro** (mRNA) e tem a seqüência complementar à seqüência de nucleotídeos do gene, com a diferença de que a base timina é substituída pela uracila.

O processo de transcrição descrito acima é válido somente para organismos procariotos (organismos que não possuem núcleo), como as bactérias e algas azuis. Nos eucariotos, os genes são compostos de partes chamadas **íntrons** e **éxons**, que se alternam dentro do gene (Figura 2.7). Após a transcrição, os íntrons são retirados do mRNA numa operação denominada **excisão** (*splicing*). Isso significa que os íntrons são partes do gene que não são utilizadas na síntese da proteína correspondente a esse gene.

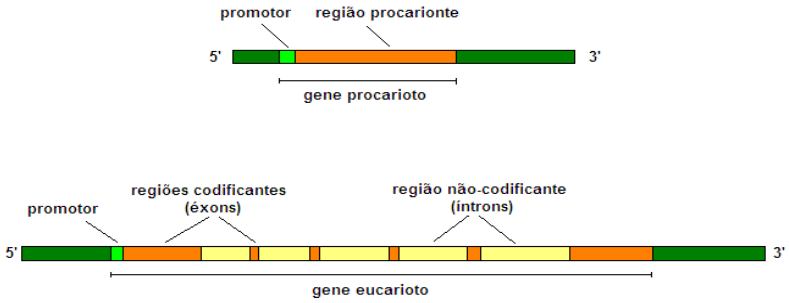

Figura 2.7: Visão esquemática dos íntrons e éxons dentro do gene.

Devido a esse fenômeno, usamos nomes diferentes para identificar o gene completo e o gene sem os íntrons. O primeiro chamamos de DNA genômico e o segundo de DNA complementar (cDNA). O cDNA pode ser

obtido a partir do mRNA através do processo chamado **transcrição reversa**. Outro fenômeno importante é a **excisão alternativa** (*alternative splicing*), que ocorre quando um mesmo DNA genômico pode dar origem a um ou mais mRNAs, através da utilização de diferentes íntrons e éxons (Figura 2.8).

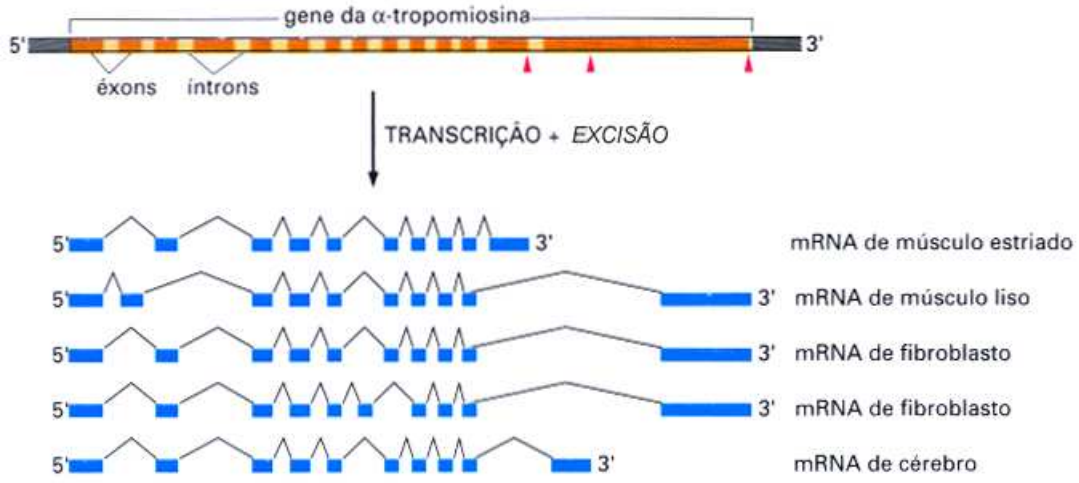

Figura 2.8: Excisão alternativa do gene α-tropomiosina de ratos (regula a contração nas células musculares).

Enquanto a transcrição se passa no interior do núcleo da célula (no caso dos eucariotos), as outras etapas do processo de síntese protéica acontecem dentro de estruturas celulares denominadas **ribossomos**, que se encontram espalhadas no citoplasma. Os ribossomos são compostos por proteínas e um tipo especial de RNA chamado de RNA ribossomal (rRNA). O ribossomo funciona como uma linha de montagem, onde as "entradas" são uma molécula de mRNA e um outro tipo de molécula de RNA chamada de **RNA transportador** (tRNA).

O tRNA é o agente encarregado do processo de tradução, isto é, a associação entre códons e aminoácidos conforme o código genético. Cada molécula de tRNA apresenta um sítio contendo um anticódon e um outro sítio onde se dará a ligação com o aminoácido codificado pelo códon correspondente (Figura 2.9). À medida que o mRNA passa pelo interior do ribossomo, ocorre uma ligação entre o códon desse mRNA dentro do ribossomo e um anticódon correspondente de um tRNA, que carrega o aminoácido apropriado. A união do tRNA com o aminoácido é facilitada por haver sempre um farto suprimento de aminoácidos no citoplasma da célula (ainda assim requer o intermédio de uma enzima — a aminoacil-tRNA sintetase (4)). A disposição

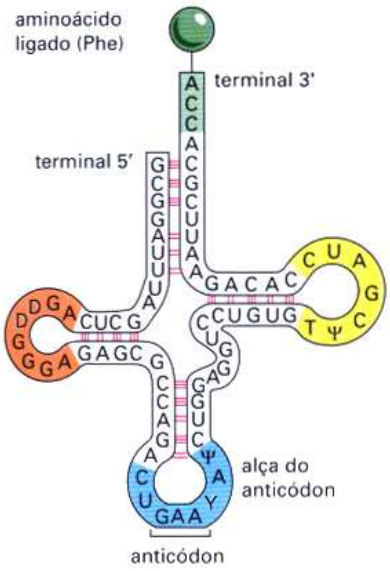

Figura 2.9: Visão esquemática da estrutura molecular de um tRNA. Observe a presença de bases não-convencionais (ψ).

tridimensional de todas essas moléculas no momento em que o tRNA se liga ao mRNA é tal que o aminoácido que está sendo adicionado é posicionado próximo ao último aminoácido da cadeia protéica que está sendo formada. O processo de desligamento desse aminoácido do tRNA e sua ligação com a proteína são catalizados pela enzima peptidil transferase (4), que faz parte do ribossomo. Quando surge um códon de terminação no mRNA, nenhum tRNA se une a ele e o processo de síntese termina. A Figura 2.10 ilustra o processo de síntese como um todo. Um mesmo mRNA poderá ser utilizado por vários ribossomos, inclusive simultaneamente (Figura 2.11), para a síntese de múltiplas cópias da mesma proteína. O tempo que cada molécula de mRNA permanecerá na célula até ser finalmente degradado varia significativamente, dependendo principalmente da seqüência de nucleotídeos que o compõem e do tipo de célula na qual foi produzido (diferentes células podem produzir uma mesma proteína em quantidades diferentes). Os processos de multiplicação do DNA, transcrição, transcrição reversa e tradução formam o que os biólogos chamam de **dogma central** da Biologia Molecular (Figura 2.12).

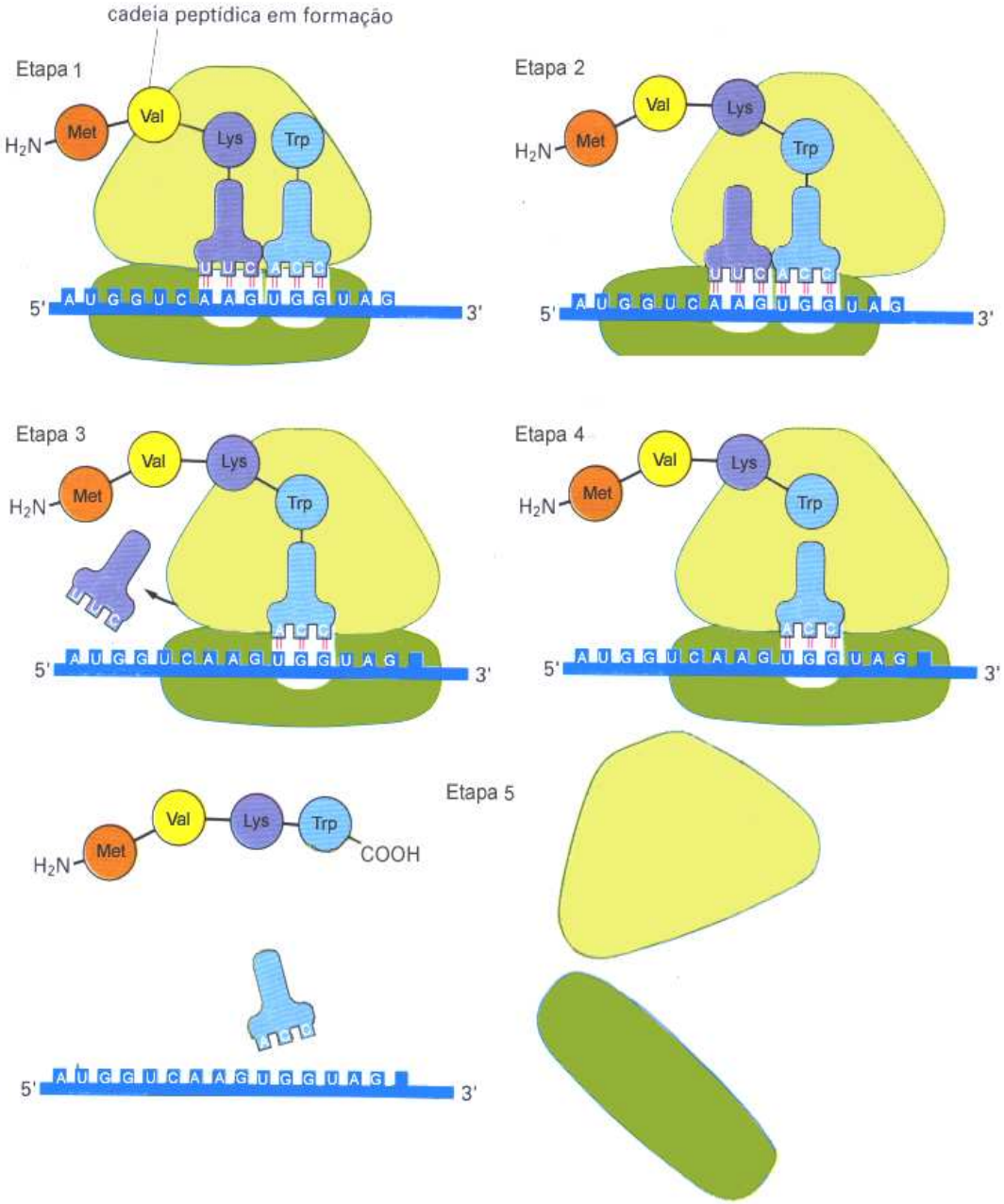

Figura 2.10: Visão esquemática do processo de síntese protéica. Etapa 1: A lisina é ligada à cadeia protéica, enquanto outro tRNA pareia com o próximo códon. Etapa 2: A lisina é desligada do tRNA, enquanto o triptofano é ligado à lisina. Etapa 3: O tRNA da lisina sai do ribossomo. Etapa 4: O triptofano é desligado do tRNA. Etapa 5: A proteína é completada e solta no citoplasma e a maquinaria de síntese (ribossomo + mRNA + tRNA) é desfeita.

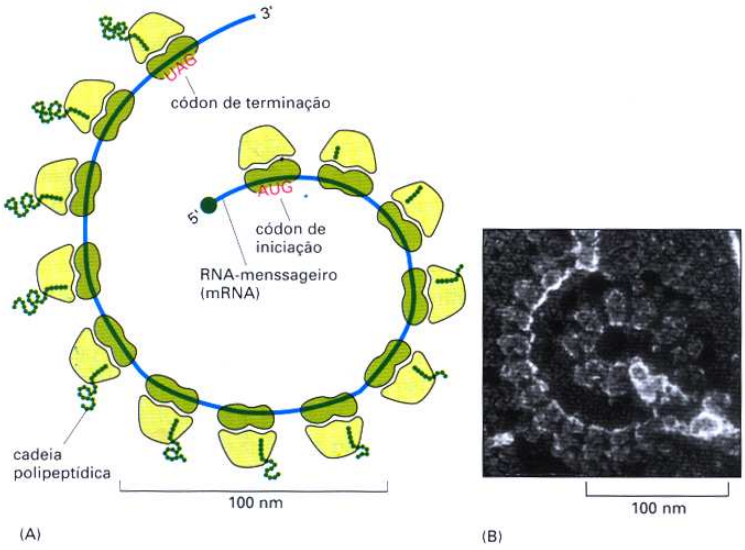

Figura 2.11: (A) Desenho esquemático mostrando vários ribossomos traduzindo uma mesma molécula de mRNA. O conjunto formado por esses ribossomos mais a molécula de mRNA é denominado polirribossomo. (B) Microfotografia eletrônica de um polirribossomo de uma célula eucariótica.

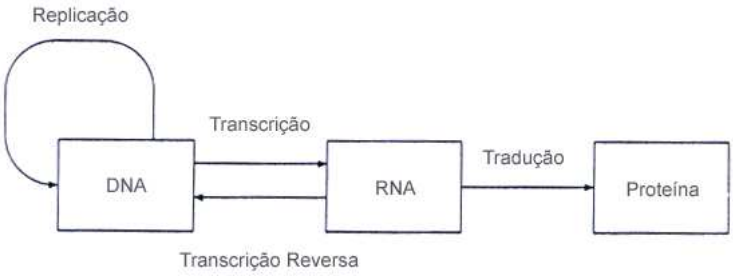

Figura 2.12: Dogma Central da Biologia Molecular (33).

2.3.2 Fases de Leitura

Um aspecto importante a ser considerado no processo de transcrição é o conceito de **fase de leitura** (*reading frame*). Uma fase de leitura corresponde a cada uma das três maneiras possíveis de agrupar códons numa seqüência de DNA ou RNA. Consideremos por exemplo a seqüência:

AGTCCGTATAATGC

Uma possível fase de leitura poderia ser obtida se considerássemos a seqüência de códons desde a primeira letra: AGT, CCG, TAT e AAT, deixando de fora GC. Uma outra possibilidade seria: GTC, CGT, ATA e ATG, eliminando uma letra de cada extremidade. Novamente, poderíamos começar apenas na terceira letra: TCC, GTA, TAA e TGC, excluindo o AG inicial. A partir da quarta letra, as fases de leituras são iguais a alguma das três primeiras com um ou mais códons a menos.

Quando se deseja contemplar a possibilidade da seqüência pertencer à fita complementar, é necessário considerar também as fases de leitura no sentido reverso complementar. Adicionam-se assim mais três possíveis fases de leitura às três que já existiam, totalizando seis possíveis fases de leitura para uma dada seqüência (Figura 2.3).

GTTGACCGTCAGCGGTGCCAACTACTG

GTT GAC CGT CAG CGG TGC CAA CTA CTG	$+1$
TTG ACC GTC AGC GGT GCC AAC TAC TG	$+2$
TGA CCG TCA GCG GTG CCA ACT ACT G	$+3$
CAG TAG TTG GCA CCG CTG ACG GTC AAC	-1
AGT AGT TGG CAC CGC TGA CGG TCA AC	-2
GTA GTT GGC ACC GCT GAC GGT CAA C	-3

Tabela 2.3: Um seqüência de DNA com as seis possíveis fases de leitura

Uma **fase aberta de leitura** (*open reading frame* ou ORF) é uma seqüência de códons que começa em um códon de iniciação e tem um número múltiplo de 3 de bases. Como conseqüência, uma ORF é completamente mapeada em códons sem que sobre nenhuma base ao final da seqüência.

2.3.3 RNA Não-Codificador (ncRNA)

Como exposto anteriormente, os genes são porções do DNA que codificam proteínas. No entanto, existem regiões na molécula de DNA que não

correspondem a genes, logo pode-se dizer que DNA é composto por regiões gênicas e regiões intergênicas. Sabe-se que cada gene, ou grupo de genes, é envolto por trechos que regulam a transcrição e outros processos correlatos (33). Além desses trechos reguladores, o DNA intergênico possui trechos responsáveis pela codificação de certos tipos de RNA (13). Atualmente funções de RNAs que participam de funções diferentes da síntese de proteínas são objeto de intensas pesquisas. Esses RNAs são chamados de **RNAs não-codificadores (ncRNA)** (33).

As mutações que ocorrem no DNA não-codificador normalmente não são letais, pois não afetam genes nem as regiões reguladoras que os envolvem.

A porcentagem de DNA não-codificador nos cromossomos varia de espécie para espécie. Procariotos tendem a ter muito pouco, sendo os seus cromossomos quase totalmente preenchidos por genes. Já os eucariotos costumam ter bastante. Estima-se que 90% do DNA humano seja composto de DNA não-codificador.

2.4 Estudando o Genoma

A informação básica que se deseja extrair de uma molécula de DNA é sua seqüência de pares de bases. O processo através do qual essa informação é obtida denomina-se **seqüenciamento**.

Nos projetos de seqüenciamento de genomas o primeiro problema que surge é o fato das técnicas utilizadas atualmente nos laboratórios só permitirem o seqüenciamento de fragmentos com cerca de 1000 pares de base, enquanto, por exemplo, um cromossomo humano tem em torno de 10^9 pares de bases (33). Assim, torna-se necessário quebrar o DNA em fragmentos menores, seqüenciá-los e remontá-los de modo a obter a seqüência original. A montagem dos fragmentos, em especial, é um problema bastante complexo e requer a utilização de técnicas da área de Matemática e Computação. Atualmente são utilizadas duas técnicas para quebrar as noléculas de DNA, são elas:

- nucleases de restrição: faz o reconhecimento de sítios específicos dentro da molécula, determinados por uma curta seqüência de bases, e quebra o DNA nesses pontos, sendo que cada nuclease é específica para uma dada seqüência de bases;

- shotgun: consiste em uma solução contendo DNA purificado — uma grande quantidade de moléculas idênticas — que é submetida a uma alta carga de vibrações (ou algum procedimento que induza à quebra desordenada das moléculas). Cada molécula é quebrada aleatoriamente em vários locais, sendo os fragmentos resultantes filtrados e separados para posterior processamento.

Para realizar qualquer experimento laboratorial com DNA é necessário uma porção mínima de material. Assim, é essencial que se possa produzir esse material em quantidade, de modo a permitir que o experimento seja repetido ou que novos experimentos sejam conduzidos. Um modo de se obter cópias de uma amostra de DNA é utilizar o mecanismo de reprodução natural de certos organismos. Primeiramente, insere-se a amostra no DNA do organismo, que é chamado de **hospedeiro** ou **vetor**. Quando o vetor se multiplica por sua reprodução natural, a amostra é replicada juntamente com o restante de seu DNA. O DNA obtido dessa forma é chamado de **DNA-recombinante**. Como o genoma do vetor já é conhecido, basta retirar essas seqüências do DNA resultante para conseguir as cópias das amostras.

Uma outra maneira de produzir cópias de moléculas de DNA é utilizar a enzima DNA polimerase. Essa enzima catalisa o processo de construção da segunda fita do DNA a partir de uma fita inicial solta, esse método é conhecido como ***Polymerase Chain Reaction*** (PCR).

2.4.1 Seqüenciamento

O seqüenciamento, isto é, a identificação das bases que compõem uma molécula ou porção de DNA, é efetuado por meio de uma técnica denominada **eletroforese em gel**. Primeiramente, através de uma replicação controlada, são gerados separadamente para cada base todos os fragmentos possíveis que terminam com aquela base(Tabela 2.4).

TGAACTGCCACAGT

TGA	T	TG	TGAAC
TGAA	TGAACT	TGAACTG	TGAACTGC
TGAACTGCCA	TGAACTGCCACAGT	TGAACTGCCACAG	TGAACTGCC
TGAACTGCCACA			TGAACTGCCAC

Tabela 2.4: Cada coluna indica todos os fragmentos da seqüência original terminados em uma base particular.

A seguir, são preparados quatro blocos de gel fino — um para cada

base — onde serão depositados os fragmentos correspondentes (por exemplo, os fragmentos terminados em G serão depositados no bloco da base G). Aplica-se então um campo elétrico nos blocos. Uma vez que o DNA é carregado negativamente, os fragmentos migram em direção ao eletrodo positivo, de tal forma que os fragmentos maiores migram mais lentamente pois o seu avanço é mais lento no gel. No decurso de várias horas, os fragmentos de DNA ficam espalhados ao longo do bloco de acordo com o tamanho, formando uma escada de faixas discretas, cada uma composta de uma coleção de moléculas de DNA de comprimento idêntico. Utilizando-se algum método para tornar possível a visualização dos quatro blocos com as faixas lado a lado, pode-se identificar a seqüência de bases que compõem o fragmento original a partir dessas faixas (Figura 2.13). Esse processo é chamado de **corrida**.

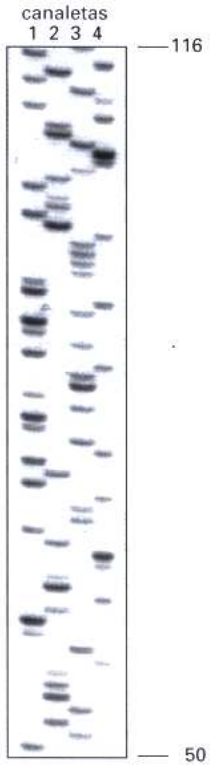

Figura 2.13: Resultado da eletroforese em gel em filme fotográfico.

Às vezes, essa técnica apresenta algumas falhas, pois as marcas podem estar borradas ou não muito claras, especialmente nas proximidades das bordas. Além disso, o tamanho dos fragmentos que podem ser seqüenciados dessa maneira está limitado a cerca de 1000 bases.

Esse trabalho, há poucos anos, era realizado manualmente pelos biólogos. Atualmente, existem seqüenciadores automáticos capazes de seqüenciar uma grande quantidade de amostras de DNA simultaneamente. O *MegaBace*, por exemplo, é capaz de seqüenciar 96 amostras em uma corrida.

A técnica utilizada pelos seqüenciadores modernos utiliza o mesmo princípio da eletroforese em gel, mas com algumas diferenças importantes. A primeira é que os fragmentos são depositados em um único bloco, chamado de capilar, ao invés de um bloco para cada base. A segunda é que a visualização dos resultados é feita através de um **eletroferograma**. O eletroferograma gerado pelo seqüenciador apresenta quatro gráficos coloridos — cada um correspondendo a uma das quatro bases. Quando uma base é identificada em uma dada posição do fragmento, o gráfico daquela base apresentará um pico na posição correspondente. Assim, a identificação no eletroferograma da seqüência dos picos e suas respectivas cores revela a seqüência de bases do fragmento seqüenciado (Figura 2.14).

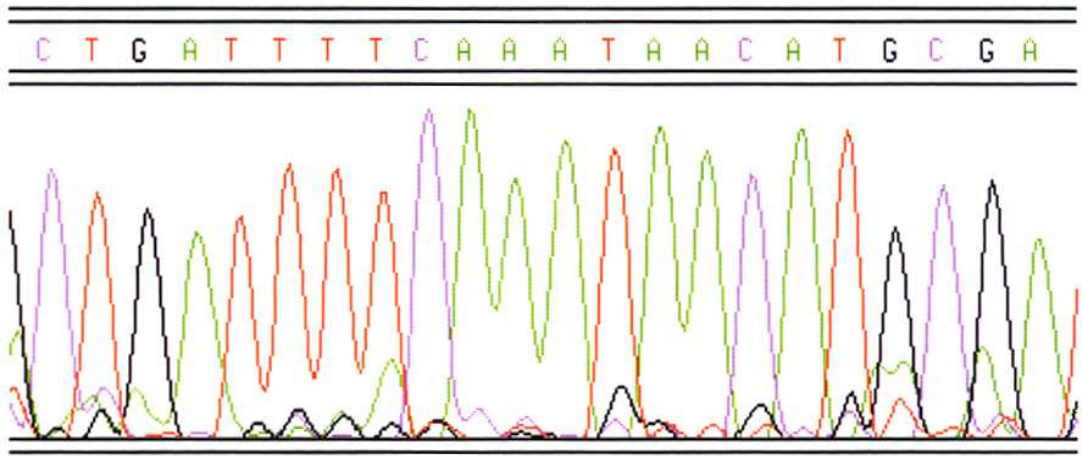

Figura 2.14: Exemplo de eletroferograma gerado por um seqüenciador automático.

2.4.2 Genoma Estrutural e Funcional

Nos projetos de seqüenciamento realizados atualmente, existem dois possíveis caminhos para analisar o DNA de um organismo: projeto **genoma estrutural** ou projeto **genoma funcional**. Os projetos de genoma estruturais visam seqüenciar todo o DNA do organismo, incluindo as partes

que não codificam proteínas (íntrons e regiões intergênicas). Já nos projetos de genoma funcional, o enfoque está no seqüenciamento das regiões gênicas cujos transcritos efetivamente farão parte do mRNA final que será traduzido nos ribossomos. O objetivo principal nos projetos genoma funcionais é conhecer os genes do organismo estudado que são expressos.

Tanto os projetos genoma estruturais como os funcionais têm dedicado uma parte do esforço ao seqüenciamento de **ESTs** (*Expressed Sequence Tags*). O conceito de EST foi proposto no final dos anos 80 (41). Uma EST é uma seqüência obtida do mapeamento de uma porção do cDNA, gerado pela transcrição reversa do mRNA citoplasmático, e pode ser feito a partir da extremidade $5'$ (EST $5'$) e/ou da extremidade $3'$ (EST $3'$) (3). O esquema de produção de ESTs pode ser observado na Figura 2.15.

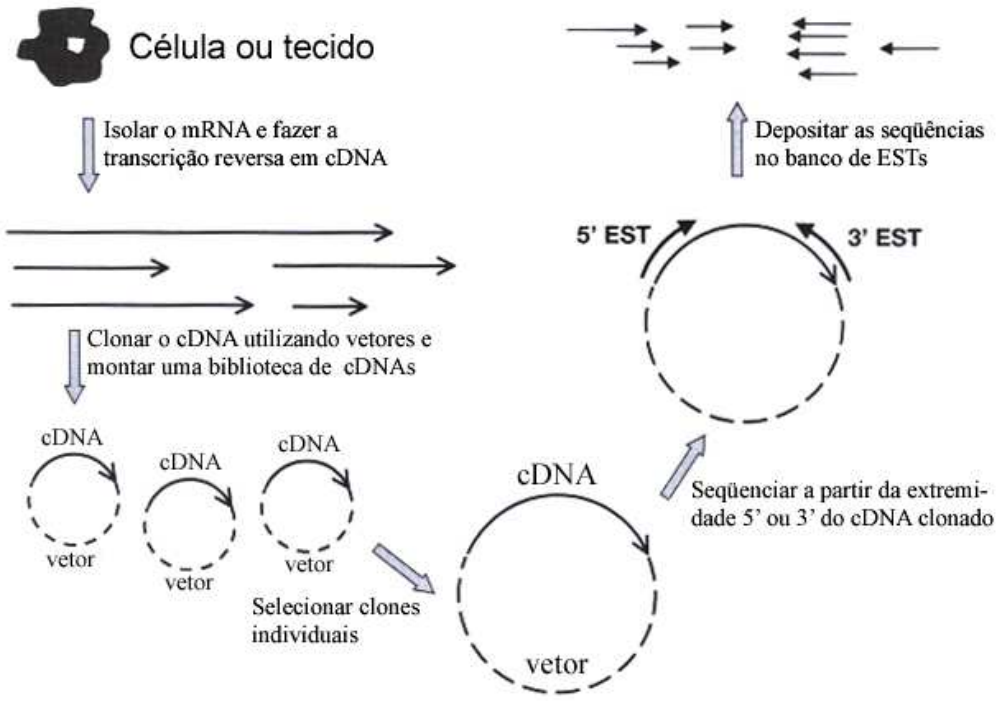

Figura 2.15: Ciclo de produção deESTs.

Geralmente ESTs têm tamanho pequeno, variando em média de 200 a 800 pares de bases e representam fragmentos de genes e não seqüências completas. Geralmente ESTs são seqüenciadas uma única vez. Por isso, as seqüências podem apresentar uma maior freqüência de erros do tipo remoções, substituições e inserções de bases quando comparadas ao mRNA original (40). Além disso, essas seqüências comumente apresentam con-

taminações com material genético do vetor, que precisa ser detectado e retirado antes, de prosseguir na análise final dessas seqüências, mas isso pode ser feito por métodos computacionais.

O seqüenciamento usando ESTs é uma estratégia rápida e eficiente na prospecção de novos genes, pois permite a análise da informação efetivamente expressa por estes, abordando principalmente o estudo de proteínas sem a necessidade de enfocar elementos estruturais ou de regulação gênica. Nesta estratégia, é possível trabalhar com pequenas seqüências, dispensando o esforço e os custos de efetuar a montagem completa do genoma. Entretanto, o conhecimento de uma seqüência genômica completa às vezes torna-se necessário, pois provê informações específicas, que nem sempre podem ser obtidas apenas com o seqüenciamento de ESTs.

Desde a descrição original do projeto de seqüenciamento de 609 ESTs, em 1991 (3), tem havido um acentuado crescimento no número de ESTs depositadas nos bancos públicos de seqüências. Em meados de 1995 o número de ESTs no *GenBank* (2) ultrapassou o número de não-ESTs. Já em junho de 2000 os registros chegaram a $4,6$ milhões de ESTs, constituindo 62% do total de seqüências depositadas no GenBank. Apesar do fato das ESTs originais serem de origem humana, o banco dbEST do NCBI (ncb) contém atualmente ESTs de mais de 250 organismos (41).

Capítulo 3

Conceitos Básicos de Bioinformática

Até o final dos anos 80, as metodologias manuais utilizadas para gerar seqüências de DNA exigiam um tempo muito maior que os seqüenciadores automáticos usados atualmente. O crescente avanço da tecnologia tem permitido que os laboratórios de Biologia Molecular forneçam detalhes cada vez mais precisos sobre as estruturas estudadas. O enorme volume de informações acumulado desde então e a necessidade de trabalhar esses dados eficientemente criou uma série de problemas que são, por natureza, interdisciplinares. Em particular, as teorias da matemática e da computação tornaram-se fundamentais no processo de manipulação de dados científicos dentro da Biologia Molecular (33).

É nesse contexto que surge a **Bioinformática**, uma nova área do conhecimento que visa estudar e aplicar técnicas e ferramentas computacionais na organização e interpretação das informações associadas às estruturas estudadas na Biologia Molecular (26). A Bioinformática, apesar de ser um campo relativamente novo, tem evoluído dramaticamente nos últimos anos e, hoje, é fundamental para compreender vários aspectos dentro da Biologia Molecular (16).

Há três objetivos principais dentro da Bioinformática. O primeiro é prover um meio de organizar os dados biológicos de forma a tornar fácil o acesso às informações e a submissão de novos dados à medida que estes são gerados. Diversos bancos de dados biológicos foram e continuam sendo criados com esse objetivo. O segundo objetivo é o desenvolvimento de ferramentas para ajudar a analisar esses dados armazenados. Por exemplo, depois de seqüenciar uma determinada proteína é interessante compará-la

com outras seqüências já produzidas em busca de alguma similaridade estrutural ou funcional. O terceiro objetivo é o processamento dos resultados gerados por essas ferramentas para analisar e interpretar as informações de uma maneira que seja biologicamente consistente e relevante (26).

Tradicionalmente, pesquisas em Biologia investigavam um sistema isoladamente e o comparava com outros poucos sistemas relacionados. A Bioinformática permitiu conduzir análises globais envolvendo um volume muito maior de dados, de forma a descobrir princípios comuns e características importantes mais facilmente (26).

3.1 *Pipeline* de um projeto de seqüenciamento de genoma

O *Pipeline* computacional para um projeto de seqüenciamento de genoma refere-se a uma seqüência de passos a serem seguidos para a realização do processamento de seqüências biológicas. De forma genérica este *pipeline* é constituído de três grandes fases: Submissão, Montagem e Anotação, que são executadas em seqüência, sendo que os dados resultantes de cada fase são utilizados como dados de entrada da fase seguinte.

Na Figura 3.1, apresentamosas três fases principais que configuram o pipeline geral.

Figura 3.1: As três grande fases de um *Pipeline* para um projeto de seqüenciamento de genoma.

Submissão

Cada fragmento do DNA (também chamado *read*) é colocado em um espaço específico e marcado de uma placa contendo vários fragmentos, de tal forma que esta placa será processada por algum sequenciador automático (por exemplo, o MegaBace). O sequenciador automático gera um arquivo de eletroferograma, que contém gráficos que identificarão cada uma das bases nitrogenadas da *read* este arquivo de probabilidade de error é gerado pelo programa *phred*

O início da etapa de Submissão ocorre quando os biólogos enviam arquivos compactados com os eletroferogramass.

A Figura 3.2 mostra um exemplo da fase de submissão, para o Projeto Genoma *Paracoccidioides brasiliensis* - Pb.

Para submeter os arquivos com todas as seqüências de uma placa, o pesquisador deve informar seu nome (usuário) e senha, para que o sistema possa realizar a autenticação. Caso a autenticação seja confirmada, o sistema criará uma sessão e o pesquisador terá acesso à todas as informações do sistema. Na submissão, o pesquisador envia o arquivo compactado (formato zip, por exemplo). Quando a transferência é concluída, inicia-se a análise. O sistema descompacta o arquivo e executa o programa *Phred* que transforma os gráficos em *strings* e calcula a qualidade das bases, gerando arquivos com extensão *phd*. O sistema executa em seguida um programa chamado *Phd2fasta*, que converte os arquivos *phd* em arquivos no formato *fasta*.Observamos que o *fasta* é um padrão utilizado em projetos de seqüenciamento e possui um formato como a apresentada na Figura 3.3.

Após a execução do *Phd2fasta*, o sistema executa o *Crossmatch* para retirar os vetores da seqüência. Neste ponto termina a etapa de submissão, sendo gerados dois arquivos, um contendo a seqüência de caracteres correspondentes às bases do fragmento seqüenciado e outro contendo as qualidades de cada base dessa seqüência. São gerados também relatórios de acordo com as necessidades do projeto.

Montagem

O processo de montagem consiste em tentar agrupar as seqüências que foram aceitas na etapa de submissão, de tal forma que seqüências contendo extremidades similares são unidas no mesmo conjunto. São processados dois arquivos de entrada, um contendo as seqüências e outro contendo as qualidades e o programa de montagem das seqüências gera como saída um grupo de *contigs* e outro de *singlets*. *Contigs* são seqüências formadas a partir de um grupo contendo duas ou mais seqüências e *singlets* são formados por um única seqüência que não foi agrupada com nenhuma outra. A Figura 3.4 mostra um exemplo da etapa de montagem, para o Projeto Genoma Pb.

Anotação

Esta etapa é dividida em anotação automática e anotação manual.

O processo de anotação automática executa programas que comparam

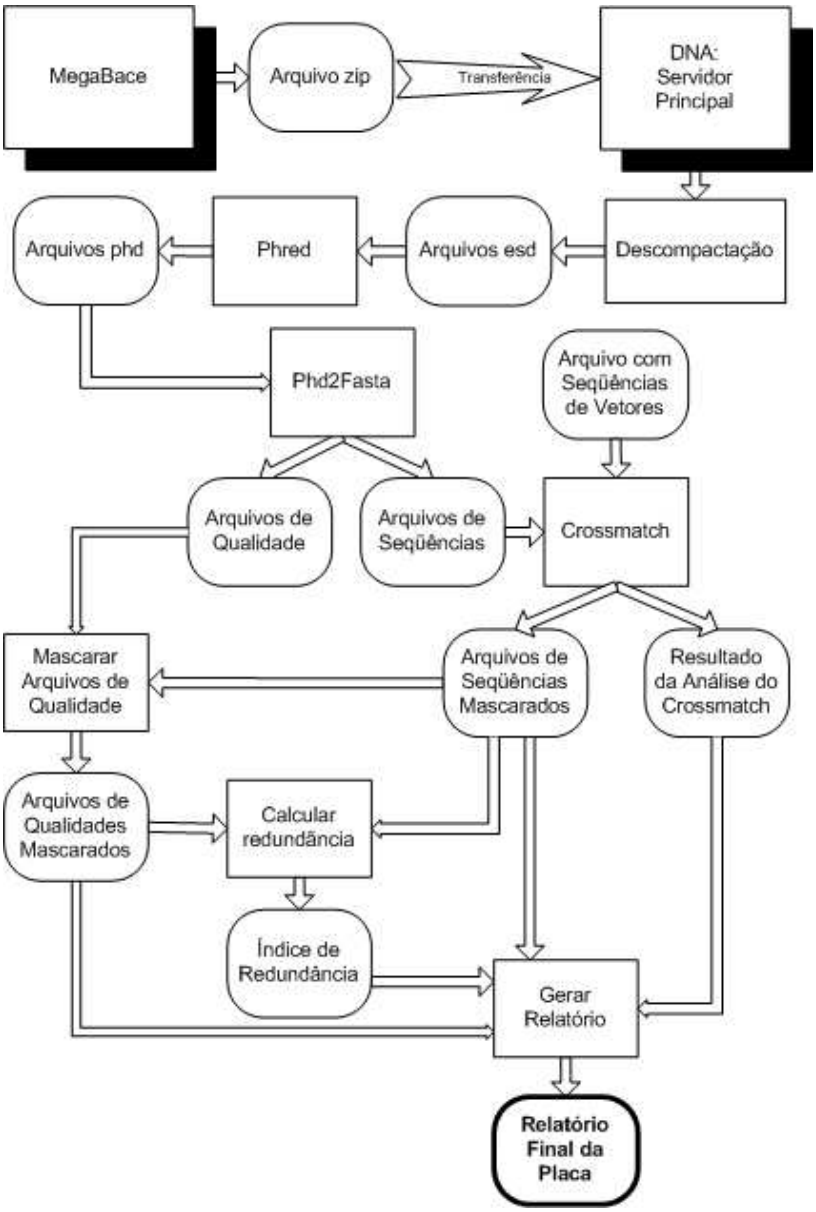

Figura 3.2: Exemplo de *Pipeline* da etapa de Submissão do Projeto Pb.

```
TGATCCCCGGCTGCAGGAATCGGCACGAGGCCCCCGATATAATATATACAGGCTTTTCAT
TCATAATTCTATCATTCATTCCATAAGACCATTCCTGTTAACACTTCTATCCAACCGCGA
CGGCCACGCCCATTGTCACAAACATTTTGGCGTCAGAGAGAACCTTAGGCGTGCATTGTG
TTCTGGCAATTGCAGTCTTCTCCCTCCTGCTTCACCACTTTAACAACATAAAATTATCCT
TTCAATATCCGTCATTATATAACCAAATGCCCCGCTTACCTATATTCATATCCTTGAAGA
```

Figura 3.3: Exemplo de um arquivo no formato fasta. A característica principal é que a primeira linha possui o caractere especial > seguido por um texto e as próximas linhas contem a seqüência.

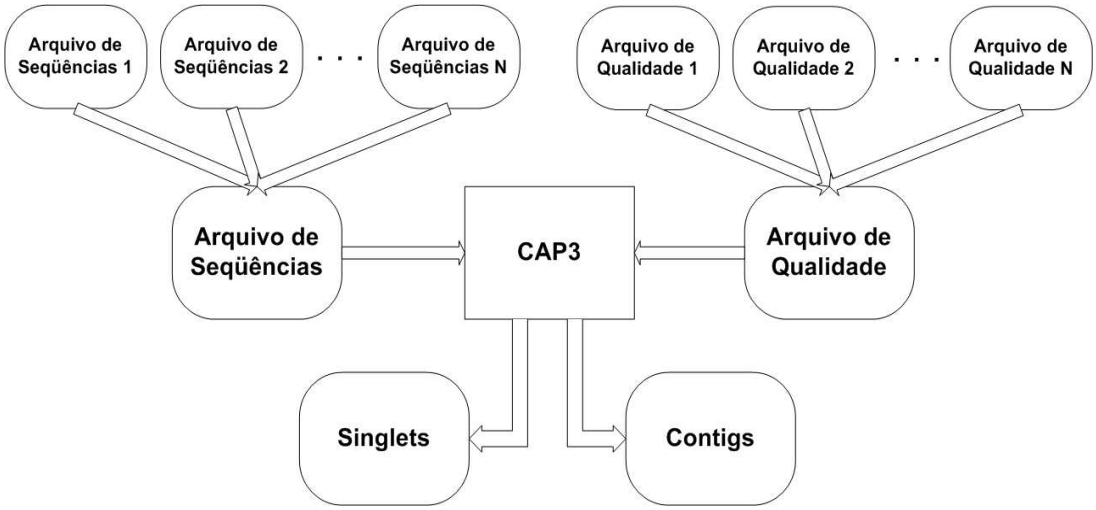

Figura 3.4: Exemplo de *pipeline* da etapa de Montagem do Projeto Genoma Pb.

as seqüências geradas no projeto com seqüências armazenadas em bancos de dados que já tiveram suas funções determinadas.

A anotação manual realizada pelo biólogo, por interface web amigável que decide a função associada a cada gene

A Figura 3.5 mostra um exemplo do processo de anotação para o Projeto Genoma Pb.

3.2 Anotação Genômica

Conforme dito acima o processo de interpretar os dados gerados em um projeto de seqüenciamento, tornando-os informações biologicamente relevantes, é chamado de anotação (24). Com o extraordinário progresso das técnicas biológicas de seqüenciamento, as etapas de anotação, correção e verificação da relevância das informações têm sido um fator limitante na maioria dos projetos de seqüenciamento. Embora a anotação de um genoma completo forneça uma perspectiva geral sobre este, não descreve detalhadamente todos os genes. Para o aprimoramento do processo de descrição dos genes, inúmeros programas foram e vêm sendo desenvolvidos, visando, por exemplo, a predição de genes, elementos regulatórios, sítios de iniciação da tradução, localização de motivos em seqüências, entre outros (29). É importante mencionar que, além da automatização da análise dos dados, uma visão biológica é essencial na interpretação e correção das informações (7).

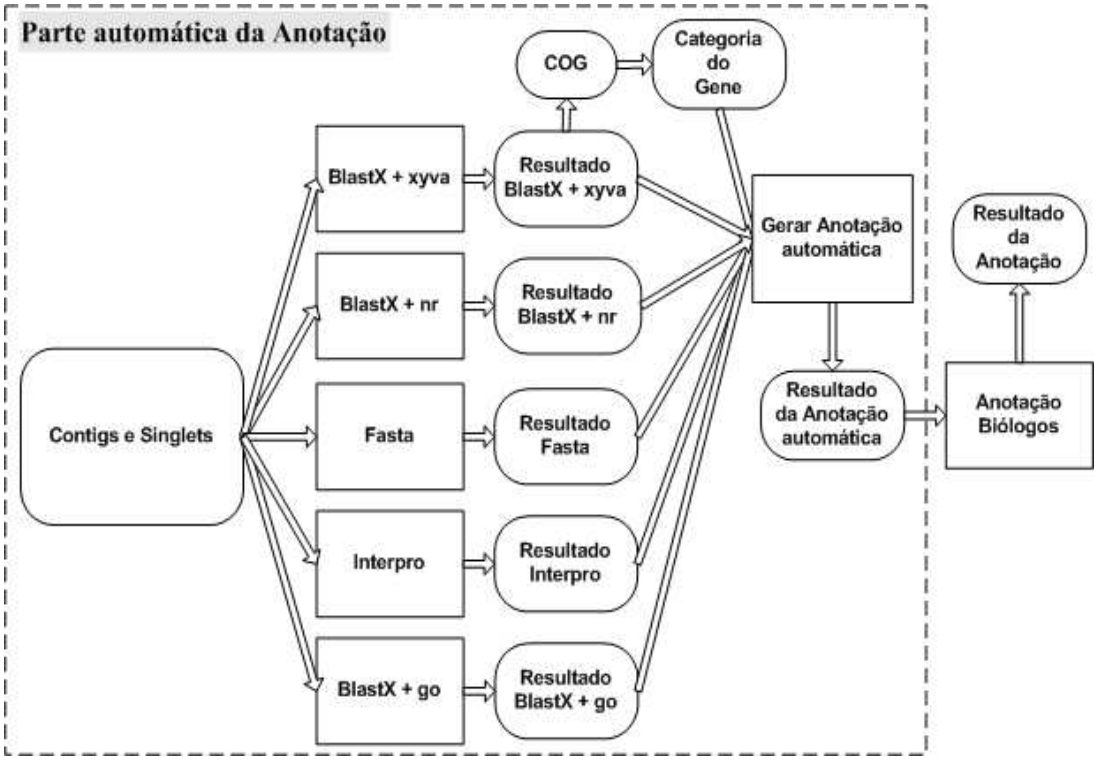

Figura 3.5: Exemplo de *pipeline* da etapa de Anotação do Projeto Genoma Pb.

Como dito anteriormente, essa etapa dos projetos de seqüenciamento é parcialmente feita no computador, por meio de programas de análise e de consulta a bancos de seqüências destinados à anotação genômica, e parcialmente feita pelos biólogos. Por isso, essa etapa é denominada **anotação semi-automática**.

Os programas e bancos de seqüências utilizados na anotação variam de acordo com as necessidades de cada projeto de seqüenciamento. Nesta seção descreveremos algumas ferramentas comumente utilizadas.

3.2.1 Ferramentas de Análise

Nesta seção, apresentaremos inicialmente programas para comparar seqüências geradas nos projetos com seqüências armazenadas nos banco de dados com informações biologicamente relevantes.

BLAST

BLAST (Basic Local Alignment Search Tool) é uma família de ferramentas para análise de similaridade entre seqüências é está entre os programas mais usados em pesquisa de bancos de seqüências no mundo (33).

BLAST utiliza uma heurística que tenta otimizar uma medida de similaridade específica. A complexidade espacial do algoritmo utilizado é expressa pelo produto do comprimento da seqüência a ser consultada (m) e comprimento das seqüências pertencentes a base de seqüências(n), ou seja, $O(mn)$ (6).

BLAST pesquisa pequenos pares de subseqüências, uma da seqüência a ser consultada e outra de uma das seqüências do banco, cujo alinhamento tem qualidade mínima de acordo com informações do usuário. Alguns desses alinhamentos são ligados para produzir alinhamentos maiores, porém, mantendo a qualidade mínima. BLAST utiliza então programação dinâmica e procedimentos heurísticos para produzir os alinhamentos finais com espaços. No caso de pesquisa de seqüências de proteínas, o BLAST utiliza matrizes de substituição para pontuar os pares de resíduos alinhados (6).

FASTA

FAST é uma outra família de programas para consulta a bancos de seqüências. O primeiro programa a ser disponibilizado publicamente, o FASTP, foi projetado para pesquisar apenas seqüências de proteínas. O algoritmo básico usado pelo FASTP era comparar cada seqüência do banco com a seqüência a ser pesquisada e retornar aquelas mais similares junto com os alinhamentos e outras informações importantes. A velocidade do FASTP era devida à sua eficiência em comparar duas seqüências muito rapidamente.

Além das ferramentas de pesquisa, a família FAST inclui um programa bastante útil para estabelecer o rigor estatístico de uma pontuação. Melhorias na detecção de similaridades no programa FASTP levaram à criação do FASTA. Uma característica incorporada foi a capacidade de processar DNA assim como seqüências de proteínas. FASTA pode ser visto como uma combinação de FASTP com FASTN, um programa projetado especificamente para analisar seqüências de nucleotídeos. Um outro programa dessa família, TFASTA, compara uma seqüência de proteínas com um banco de seqüências de DNA, fazendo as traduções nas seis fases de leitura (28).

Uma outra observação relaciona-se com a computação das pontuações iniciais. 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. Com isso, as pon-

tuações iniciais melhoram significativamente para seqüências relacionadas. Além disso, as 10 melhores regiões são escolhidas em FASTA, enquanto que no FASTP eram apenas 5. Outros programas que usam as mesmas técnicas também fazem parte do pacote. LFASTA é uma ferramenta para similaridades locais, reportando mais do que um bom alinhamento entre um par de seqüências.

PFAM

O **Pfam** (pfa) é banco de alinhamentos de famílias de domínios protéicos (região de uma proteína com uma função específica). Um exemplo é a DNA polimerase, que possui um domínio responsável exclusivamente pela ligação com a fita de DNA. O Pfam é dividido em dois bancos menores: o Pfam-A e o Pfam-B . O banco Pfam-A contém mais de 2700 perfis de alinhamento (a maioria destes perfis abrange domínios protéicos inteiros), sendo que esses alinhamentos foram montados permitindo-se a inserção de espaços. Por sua vez, o banco Pfam-B é gerado automaticamente a partir do agrupamento das seqüências que sobraram durante o processo de construção do Pfam-A.

PSORT

O ***PSORT*** é um pacote de programas, desenvolvido por Nakai em 1991, para realizar a predição da localização de proteínas nas células. De acordo com o tipo de organismo ao qual a proteína pertence, um conjunto de localizações possíveis é testado. O *PSORT* verifica se a proteína pertence ao citoplasma, à membrana interna, à membrana externa ou ao periplasma da célula, um espaço situado entre a membrana externa e membrana citoplasmática (pso).

INFERNAL

O ***INFERNAL*** é um pacote de ferramentas que permite ao usuário fazer *perfis*[1] em estruturas secundárias de RNA similares ou criar novas estruturas baseadas em alinhamentos múltiplos de seqüências (12). O INFERNAL é também capaz de fazer a detecção de RNAs não codificadores.

O *INFERNAL* é comparável ao *HMMER* (Vide http://hmmer.wustl.edu/). O pacote de ferramentas *HMMER* utiliza Modelos de Markov - de seqüências lineares - para produzir *perfis* de alinhamentos múltiplos de seqüências, chamados de *perfis* HMMs. Os *perfis* HMMs capturam apenas carac-

[1] *Perfis* de alinhamento são representações numéricas, normalmente matriciais, de um alinhamento múltiplo. Esse perfil representa as características comuns àquele particular conjunto de seqüências, que é, em geral, uma família de proteínas.

terísticas de estruturas primárias similares. Os *perfis* produzidos pelo *INFERNAL* incluem informações sobre seqüências e estruturas secundárias de RNAs.

O INFERNAL é a implementação de modelos probabilísticos para a análise de estruturas de RNA, chamados de Modelos de Covariância (MC). Uma vantagem considerável desta implementação foi o ganho na complexidade do algoritmo de programação dinâmica para MC em estruturas de RNA (em relação aos baseados nos Modelos de Markov). O algoritmo base do INFERNAL é análogo ao algoritmo de Myers/Miller (para alinhamentos de seqüências lineares), porém tem uma complexidade da ordem de $O(N^2 \log N)$ enquanto outros possuem complexidade da ordem de $O(N^3)$ (11).

A ferramenta gera saídas robustas, porém com falta de muitos dados acerca de características relevantes para os especialistas. É muito importante ressaltar que os algoritmos do *INFERNAL* requerem um enorme tempo de processamento, tornando-o uma ferramenta muito lenta e que os projetos que venham a necessitar de seu uso provavelmente vão requerer o uso de vários processadores através de um ambiente distribuído, tal como computação em grade por exemplo.

Capítulo 4

Inteligência Artificial

Neste capítulo serão apresentados conceitos básicos Inteligência Artificial (IA) e de Sistema Multiagente (SMA), os quais foram utilizados no desenvolvimento deste trabalho. Na Seção 4.1 será apresentado uma visão inicial sobre representação do conhecimento, na Seção 4.2 será caracterizado agente inteligente, Sistemas Multiagente, será feita uma breve descrição do padrão FIPA e a caracterização de Ontologia, sendo destacado seu uso na Bioinformática.

4.1 Representação do Conhecimento

IA possui enumeras definições, de tal forma que não existe um consenso sobre o que ela realmente é. Pode-se dizer que a IA é uma ciência dedicada a buscar métodos ou dispositivos computacionais que possuam ou simulem a capacidade humana de resolver problemas, pensar ou, de forma ampla, ser inteligente.

Neste trabalho usamos outro conceito importante dentro da IA, o conceito de Inteligência Artificial Distribuída (IAD), que se refere a sistemas cujo o funcionamento depende de um determinado conjunto de partes menores para resolver, de modo cooperativo, um determinado problema, assim nesta área existe um foco maior no comportamento social dos agentes para a resolução de problemas.

Para que exista inteligência é necessário o tratamento de conhecimento através da aquisição, do armazenamento e de inferência que possibilitem o tratamento do conhecimento. Para que o conhecimento possa ser armazenado é preciso que se possa representá-lo. Muito esforço em IA tem se especializado em buscar ou melhorar formalismo para a representação do

conhecimento. Qualquer processo inteligente realizado por uma máquina deve conter uma estrutura que permita uma descrição proposicional do conhecimento exibido pelo processo, e que, independentemente da semântica, tenha um papel formal, causal e essencial na geração do comportamento que manifesta tal conhecimento. Vários modelos de representação foram criados, cada um visando um certo grupo de problemas. Os modelos mais utilizados são:

- Redes: Conhecimento é representado por um rótulo de grafos direcionados cujos nós representam conceitos e entidades, enquanto os arcos representam a relação entre eles. Redes Semânticas: são primitivas de representação, sendo que é a forma de representação mais adequada para domínios onde problemas podem ser descritos como taxonomias ou classificações complexas. Redes semânticas são representadas como um conjunto de nós ou nodos que são ligados por meio de arcos, onde cada nodo representa um objeto, uma entidade conceitual ou um evento e cada arco representa o relacionamento existente entre cada par de nodos, sendo que cada par de nodos representa um determinado fato;

- Lógica: Um modo de declaração que representa o conhecimento sendo uma das mais primitivas formas de representação do conhecimento humano e do raciocínio. A lógica preposicional é considerada a forma mais comum da lógica. Baseia-se em que proposição só pode assumir um dos seguintes valores: verdadeira ou falsa. A lógica de predicados é considerada como uma extensão da lógica preposicional. Na lógica de predicados os elementos fundamentais são, além do objeto, também os seus predicados;

- Frames:Frame é uma representação de um objeto complexo. Ele é identificado por um nome que consiste em um conjunto de *slots*. Cada frame possui ao menos um frame hierarquicamente superior e, portanto, constitui uma base com mecanismo de herança. Um frame especial é a raiz desta hierarquia de herança. Podemos dizer que o frame é uma coleção de atributos, chamados de *slots* com valores, que descrevem alguma entidade do mundo. Os frames integram conhecimento declarativo sobre objetos e eventos, com conhecimento procedimental, sobre como recuperar informações ou calcular valores;

- Regras de Inferência: O uso de sistemas de produção para codificar regras definidas através do esquema de condição-ação. As regras de produção, consistem em representar o domínio do conhecimento através de um conjunto de regras. Suas principais características são a modularidade, facilidade de implantação e também a grande quantidade de pacotes desta técnica existentes para o desenvolvimento de sistemas.

Neste trabalho utilizamos o modelo de Regras de Inferência, pois o problema a ser resolvido apresenta-se através de um conhecimento a ser formalizado e tratado sobre um conjunto de regras condição-ação, que, quando satisfeitas suas condições, deve-se executar determinadas ações.

4.1.1 Regras de Inferência

Inferência é um processo pelo qual se chega a uma proposição, afirmativa ou negativa baseado em uma ou mais proposições aceitas como ponto de partida do processo. Existem dois mecanismos de inferência no escopo de encadeamento das regras, o progressivo (Forward Channing) e regressivo (backward chaining).

No encadeamento progressivo, a parte esquerda da regra é comparada com os fatos contidos na base de fatos, que nada mais são que proposições verdadeiras em determinado instante de tempo. As regras que satisfazem a esta descrição têm sua parte direita executada, o que, em geral, significa a introdução de novos fatos na base de fatos, ou novas verdades. Por exemplo, temos um conjunto de condições (X, Y e Z) que compõem a parte esquerda da regra e quando essas condições forem satisfeitas então a parte direita da regra será executada(W). Sendo que W é uma ação que pode inserir um novo fato na base de fatos.

X,Y,Z -> W

No encadeamento regressivo, ocorre um processo relativamento contrário ao encadeamento progressivo, da direita para a esquerda. O comportamento do sistema é controlado por uma lista de objetivos a serem alcançados. Um objetivo pode ser satisfeito diretamente por um ou mais elementos da base de fatos, ou podem existir regras que permitam inferir algum dos objetivos correntes, isto é, que contenham uma descrição deste objetivo em suas partes direitas. As regras que satisfazem esta condição

têm as instâncias correspondentes às suas partes esquerdas adicionadas à lista de objetivos correntes. Caso uma dessas regras tenha todas as suas condições satisfeitas diretamente pela base de fatos, o objetivo/condição em sua parte direita é também adicionado à base de fatos. Um objetivo que não possa ser satisfeito diretamente pela base de fatos, nem inferido através de uma regra, é abandonado. Quando o objetivo inicial é satisfeito, ou não há mais objetivos, o processamento termina. Este mecanismo é normalmente utilizado para validar conclusões feitas em instantes te tempos diferente da verificação dos fatos constantes da base de fatos.

Neste trabalho foi utilizado somente o encadeamento progressivo devido às características de inferência do conhecimento no processo de anotação manual se assemelharem a existência de verdades para ações ou decisões de recomendações de anotações. Mas nada impede que numa extensão de implementação possamos utilizar o mecanismo regressivo para alcance de determinados objetivos no processo de anotação

4.2 Sistema Multiagente

4.2.1 Agentes Inteligentes

Um agente é uma entidade capaz de perceber o seu ambiente onde ele se insere através de sensores e de agir sobre esse ambiente através de atuadores, conforme mostrado na figura 4.1. Nesta figura, observamos a função f:p* -> A que representa a função de agente, que mapeia seqüências de percepções específicas para uma determinada ação.

Para um melhor entendimento da figura e dos termos mencionados acima, consideremos, por exemplo, um agente robótico. Neste agente poderíamos ter como sensores câmeras e detectores da faixa de infravermelho e seus vários motores, por exemplo, braços robóticos, esteiras de movimentação, funcionando como atuadores. Um outro exemplo poderia ser um agente de *software*, que recebe seqüências de teclas digitadas, conteúdo de arquivos e pacotes de rede como entradas sensoriais e atua sobre o ambiente exibindo algo na tela, gravando arquivos e/ou enviando pacotes na rede.

Um agente inteligente deve possuir capacidades como: proatividade, sociabilidade (noção de sociedade) e reatividade. Por proatividade podemos entender que os agentes inteligentes devem tomar atitudes/ações para que possam atingir seus objetivos demostrando iniciativa própria. A noção de

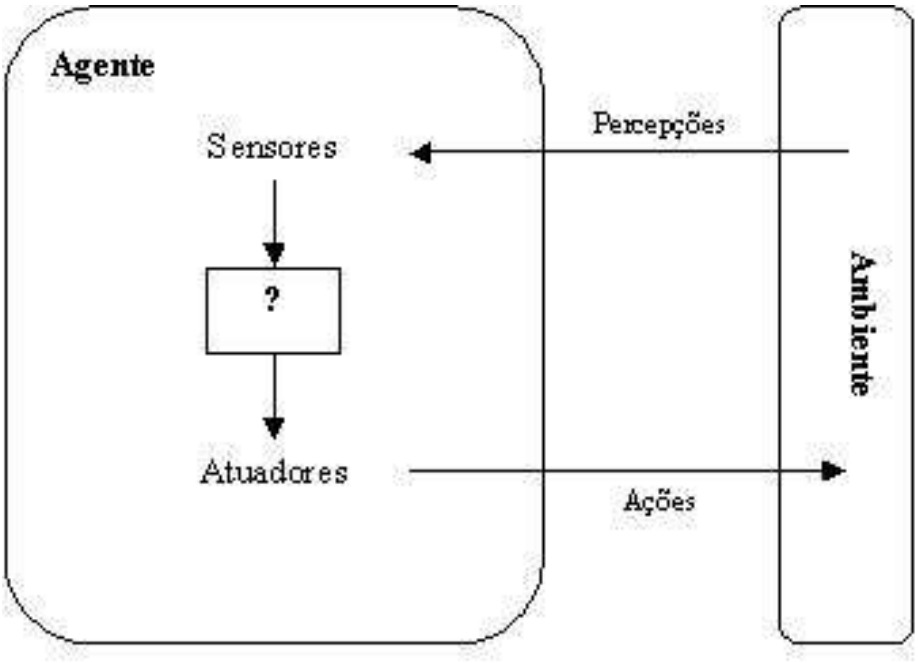

Figura 4.1: Características de um agente inteligente (30).

sociedade deve-se ao fato de que os agentes podem e devem se relacionar com os outros agentes, ou seres humanos no ambiente, por exemplo, a comunicação dentro de um SMA para alcançar seus objetivos pode ser considerada como um procedimento social de interação entre os agentes. Por fim, o conceito de reatividade está ligado à percepção do ambiente pelo agente, e sua resposta pode ser entendida em tempo hábil às alterações ocorridas, sempre agindo de forma a alcançar seus objetivos (42).

Abaixo, são relacionadas características desejáveis de um agente inteligente (35):

- execução de tarefas: o agente pode executar tarefas delegadas tanto por humanos como por outros agentes de software/hardware que pertençam ao ambiente relacionado;
- confiabilidade: o agente deve realizar com segurança e eficácia as tarefas delegadas pelo usuário;
- adaptabilidade: o agente deve se adaptar bem as alterações das necessidades dos usuários e do ambiente;
- colaboração: o agente deve colaborar tanto com os humanos quanto com os outros agentes. Esta característica permite que os agentes

incrementem seus conhecimentos, resolvam conflitos e inconsistências nas informações recebidas e nas tarefas realizadas, assim melhorando a capacidade de suporte à decisão;

- atividade: o agente deve iniciar suas atividades para solução de problemas, antecipar informações e alertar os usuários sobre as informações relevantes, além de decidir quando mesclar informações para prover um melhor entendimento.

4.2.2 Sistema Multiagente

Há vertentes, direcionadas por diversos grupos de pesquisa, em relação ao paradigma de agentes inteligentes, na qual acredita-se que a inteligência não pode ser separada do contexto social, ou seja, não se deve considerar os agentes apenas como entidades inteligentes isoladas. A partir deste contexto surge a idéia de SMA (14; 42; 15; 30).

Um SMA consiste de agentes que interagem entre si, tipicamente por envio de mensagens através de alguma infra-estrutura de rede computacional (42); ou através de *softwares* capazes de viabilizar esta comunicação, por exemplo, o *framework* JADE (9) que será explanado na Seção 4.3.1. Na Figura 4.2 é apresentada a estrutura de um agente e suas características desejáveis dentro de um SMA.

O Módulo de Conhecimento contém a informação usada pelo agente bem como os resultados das ações. Essas ações são executadas pelo Módulo de Ação que processa a execução das requisições. Essas requisições são enviadas periodicamente pelo sistema. Estas ações podem ser, por exemplo, ações *BLAST* (que executa a ferramenta NCBI *BLAST*) ou ações de detecção de genes (*Genemark, Glimmer*). O Módulo de Comunicação recebe as mensagens enviadas pelos outros agentes e pelo sistema e as envia para o destino apropriado. As requisições para execução de ações são enviadas para o Módulo de Ação enquanto as mensagens de dados são enviadas para o Módulo de Conhecimento. Os agentes trocam mensagens utilizando determinadas linguagens de comunicação, como por exemplo a *Agent Communication Language - ACL -*, padrão FIPA (27), ou a *Knowledge and Query Manipulation Language* - KQML. Os agentes que compõem um SMA podem interagir ou se comunicam entre si ou com os usuários, de acordo com seus objetivos e motivações.

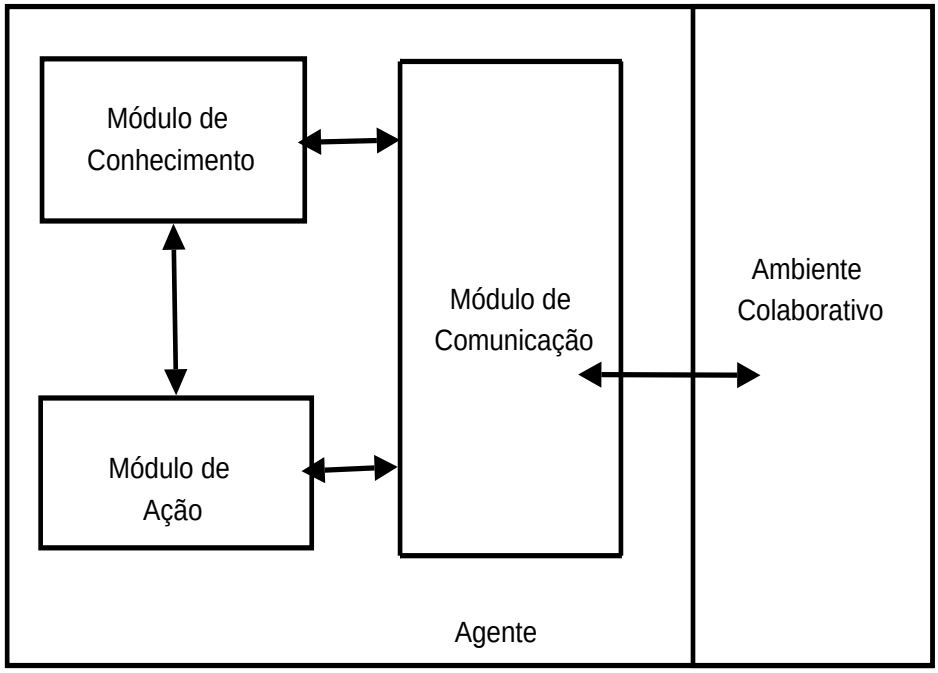

Figura 4.2: Estrutura de um agente na abordagem de um Sistema Multiagente (iwg).

Segundo Sycara (35) as características de SMA são tais que (1) cada agente possui uma visão limitada; (2) não há controle global do sistema; (3) trabalham com dados descentralizados; e (4) a computação é assíncrona (permitir comunicação entre entidades heterogêneas) .

4.2.3 FIPA

A ***F**oundation for **I**ntelligent **P**hysical **A**gents* - FIPA - é uma organização do *Institute of Electrical and Electronics Engineers* - IEEE - responsável por especificar padrões para o desenvolvimento de tecnologias baseadas em agentes inteligentes. No padrão FIPA conforme descrito em (36), 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 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 (sendo que esse atributo não é obrigatório).

A arquitetura da plataforma de agentes FIPA está contida nas especificações normativas da *Agent Management Specification* (documento que

formaliza as especificações definidas pela FIPA). A plataforma de agentes FIPA provê a infra-estrutura na qual os agentes podem ser desenvolvidos, o que estabelece um modelo de referência lógico para criação, registro, localização, comunicação, migração e retirada dos agentes. A Figura 4.3 ilustra este modelo.

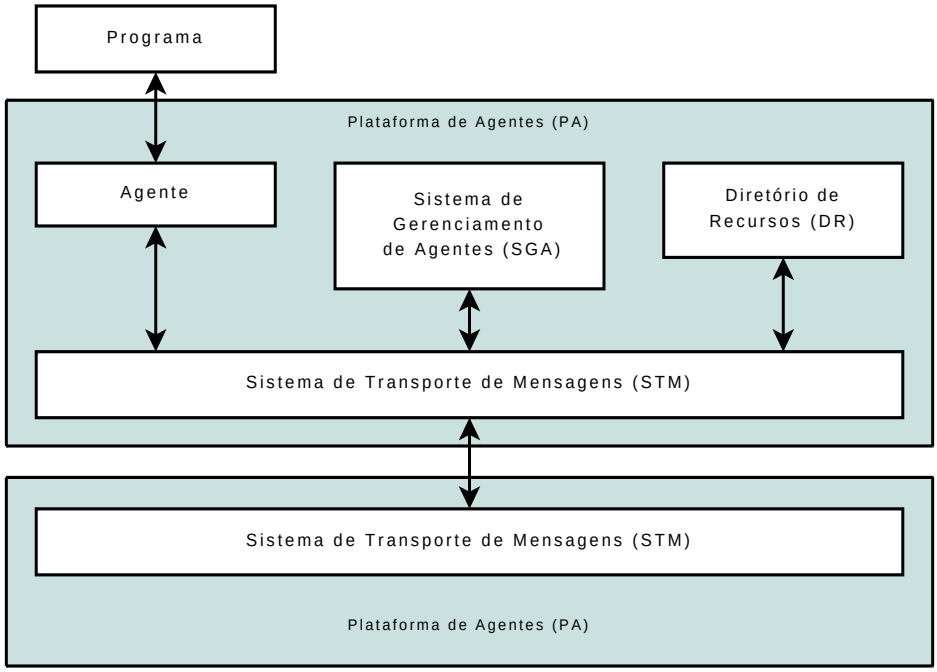

Figura 4.3: Modelo de referência para gerenciamento de agentes segundo a FIPA (36)

O modelo de referência de gerenciamento do agente é constituído dos seguintes componentes lógicos:

- O Agente - componente fundamental numa plataforma de agentes que combina um ou mais capacidades de serviços e pode incluir acesso a *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 IDentificador de Agente - IDA -, que rotula um agente, e ele pode distingui-lo dos demais sem risco de ambigüidade no seu universo.

- O Diretório de Recursos (DR) - componente obrigatório e de grande importância na plataforma de agentes. Ele provê um serviço de páginas amarelas para os outros agentes, ou seja, uma catálogo de serviços. Os

agentes podem registrar seus serviços com o DR ou requisitar ao DR que ele encontre serviços oferecidos por outros agentes. Múltiplos DRs podem existir dentro de uma mesma plataforma de agentes e podem estar confederados.

- O Sistema de Gerenciamento de Agentes (SGA) - componente obrigatório nesta plataforma de agentes. O SGA exerce um controle de acesso e uso da plataforma. Só pode existir um único SGA em uma única plataforma de agentes. O SGA apresenta um diretório de IDAs que contém endereços de transporte para agentes registrados na plataforma. O SGA oferece páginas brancas para os outros agentes, ou seja, uma catálogo de nomes. Cada agente deve se registrar com o SGA de forma a receber uma IDA válida.

- O Sistema de Transporte de Mensagens (STM) - método de comunicação padrão entre agentes de diferentes plataformas.

- A Plataforma de Agentes (PA) - provê a estrutura física em que o agente pode ser desenvolvido. A PA é composta de máquina(s), sistema operacional, *software* de suporte de agentes (exemplo, JADE), componentes de gerenciamento de agentes FIPA (DR, SGA e STM) e agentes. O mecanismo interno da PA é particular ao desenvolvedor do SMA e não há uma padronização para isso na FIPA. A FIPA está interessada somente em como a comunicação é feita entre agentes nativos da PA e fora dela ou àqueles que podem dinamicamente se registrar com uma PA. Agentes são livres para trocar mensagens diretamente utilizando quaisquer meios que eles possam suportar.

- O Programa - descreve todos os não-agentes como coleção de instruções executáveis acessíveis através de uma agente. O agente pode acessar o Programa, por exemplo, para adicionar novos serviços.

A FIPA estabeleceu um padrão pra desenvolvimento de *frameworks* de desenvolvimento de SMA o qual foi adotado pelo *framework* JADE adotado neste trabalho e apresentado na seção 4.3.1.

4.2.4 Ontologia

Segundo Gruber (19), ontologia[1] é uma especificação explícita de uma conceitualização. Neste contexto, conceitualização é a organização do conhecimento sobre o mundo em forma de entidades e a especificação é a representação da conceitualização em uma forma concreta. Para a IA ontologia significa tudo o que pode ser representado.

Almeida e Bax (5) definem características e componentes básicos comuns presentes em grande parte das ontologias. Mesmo apresentando propriedades distintas é possível identificar tipos bem definidos. Alguns componentes básicos observados em ontologias são:

- Relações: representam o tipo de interação entre os conceitos de um domínio;
- Axiomas: usados para modelar sentenças sempre verdadeiras;
- Instâncias: utilizadas para representar elementos específicos, ou seja, os próprios dados;
- Classes: organizadas em uma taxonomia[2].

Algumas das propostas definem tipos de ontologias relacionando-as à sua função, grau de formalismo de seu vocabulário, à sua aplicação e à estrutura e conteúdo da conceitualização (5).

Em relação a metodologias, muitas são construídas para sistematizar a construção e a manipulação de ontologias. Atualmente existem metodologias para construção de ontologias, construção de ontologias em grupo, aprendizado sobre a estrutura de ontologias e integração de ontologias (5).

Em Lópes (25) é apresentada uma revisão de algumas metodologias conhecidas, como a metodologia utilizada no projeto *Enterprise Ontology* (http://www.aiai.ed.ac.uk/project/) e a *METHONTOLOGY* (17). As várias metodologias existentes possuem características diversas. Não é trivial a unificação das propostas em uma única metodologia. Segundo Almeida e Bax (5), para verificar a utilidade das metodologias e compará-las, é necessário avaliar a ontologia resultante da aplicação de cada metodologia.

[1] O termo ontologia conforme a Filosofia, significa uma sistemática da existência.

[2] 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.

4.2.5 Ontologia aplicada à Bioinformática

Em banco de dados heterogêneos, por exemplo, os bancos de dados biológicos, o grande volume de dados é devido ao fato de que cada área da biologia gera seus próprios dados e acabam desenvolvendo soluções particulares para armazená-los. Para tratar com o problema da heterogeneidade faz-se necessário entender e definir os conceitos inerentes a este contexto e isto pode ser feito através do uso de ontologia (22).

Segundo Lemos (22), as ontologias eliminam incertezas e falhas de interpretações sobre os significados dos banco de dados, programas e seus relacionamentos, sendo assim, o seu uso facilita a criação de sistemas aplicados à Bioinformática. Dois tipos de ontologias podem ser destacadas a partir do cenário descrito.

- Ontologia de Biologia Molecular: identifica e associa os conceitos da Biologia Molecular. Exemplos: Gene Ontology (GO) (go), TaO (Tambis Ontology) (8), OMB (*Ontology for Molecular Biology*) (31; 32), entre outros;

- Ontologia de Processos de Bioinformática: define conceitos como as entradas, saídas e utilização de cada programa na análise de dados da Biologia Molecular.

Em relação à codificação das ontologias, elas tipicamente são codificadas usando uma forma declarativa de representação do conhecimento, que pode ser através do uso de lógica de primeira ordem ou suas variantes (20). As ontologias são úteis, no domínio da Biologia Molecular, para prover representações alternativas dos dados de diferentes perspectivas ontológicas ou para integrar dados de múltiplas fontes.

Neste trabalho foi definido uma ontologia de aplicação envolvendo as classes organizacionais em um taxonomia relacionada ao BLAST, FASTA e comunicação de agentes.

4.3 Ferramentas para SMA

Nesta Seção apresentaremos ferramentas para o desenvolvimento de SMA, em específico o *framework* de desenvolvimento utilizado JADE, as ferramentas do projeto BIOJAVA, o motor de inferência JESS e a ferramenta de otologias Protégé.

4.3.1 JADE

Esta seção descreve o *framework* escolhido para o desenvolvimento do SMA proposto neste trabalho. O *Java Agent DEvelopment Framework* - JADE -, que segue os padrões da organização FIPA e tem sido amplamente utilizado em vários projetos como: projetos que trabalham com ontologias; projetos na área de telecomunicações e projetos voltados para computação em grade.

Segundo Bellifemine (9), o *framework* **JADE**, desenvolvido pela Telecom Italia LAB - TILAB- , é um *middleware* utilizado para desenvolvimento e execução de aplicações *peer-to-peer* baseadas em agentes inteligentes, que pode operar tanto em ambientes computacionais *wired* como *wireless*. JADE é um *framework* completamente desenvolvido em Java e está fundamentado de acordo com os seguintes princípios:

- Interoperabilidade: Conformidade com a especificação FIPA (36). Isto significa que os agentes construídos com o JADE podem interoperar com outros agentes desde que estes obedeçam a especificação FIPA.
- Uniformidade e Portabilidade: JADE provê um conjunto homogêneo de APIs [3] que é independente da rede e da versão Java utilizada. Em relação a linguagem Java, as APIs JADE fornecem recursos para desenvolvedores em ambientes J2EE[4], J2SE[5] e J2ME[6].
- Facilidade de Uso: Apesar do *framework* possuir uma determinada complexidade, esta fica transparente ao usuário devido a simplicidade provida pelo seu conjunto de APIs.
- Filosofia *"Pay-as-you-go"*: Os desenvolvedores não precisam usar todos os recursos fornecidos pelo *middleware*. Os recursos que não estão sendo utilizados, não trazem *overhead* aos recursos computacionais.

O JADE possui uma vasta biblioteca, através de um conjunto de classes Java, para o desenvolvimento de agentes inteligentes, bem como um am-

[3] *Application Program Interfaces*, é um conjunto de rotinas, classes, protocolos e ferramentas para construção de *softwares*

[4] ***J****ava 2 Platform* ***E****ntreprise* ***E****dition* é um ambiente de desenvolvimento para aplicações máquinas de grande porte como, por exemplo, em servidores de grande capacidade.

[5] ***J****ava 2 Platform* ***S****tandard* ***E****dition* é um ambiente de desenvolvimento para aplicações em *desktop*.

[6] ***J****ava 2 Platform* ***M****icro* ***E****dition* é um ambiente de desenvolvimento para aplicações em redes *wireless* e dispositivos móveis, por exemplo.

biente de execução que provê serviços básicos, os quais devem ser ativados no dispositivo antes dos agentes serem executados[7]. Cada instância, em tempo de execução do JADE, é chamada de *container* (desde que esta contenha agentes). O conjunto de *containers* é chamado de plataforma e provê uma camada homogênea a qual não é transparente aos agentes, pois o *container* determina a localização de um agente na plataforma.

O *framework* JADE é extremamente versátil, com bom desempenho tanto em conjunto com aplicações construídas em arquiteturas complexas, tais como .NET or J2EE, ou em conjunto com aplicações executadas em ambientes com recursos limitados, tais como dispositivos móveis, como telefones celulares. Na Figura 4.4 temos a arquitetura do *framework* JADE apresentada em (9).

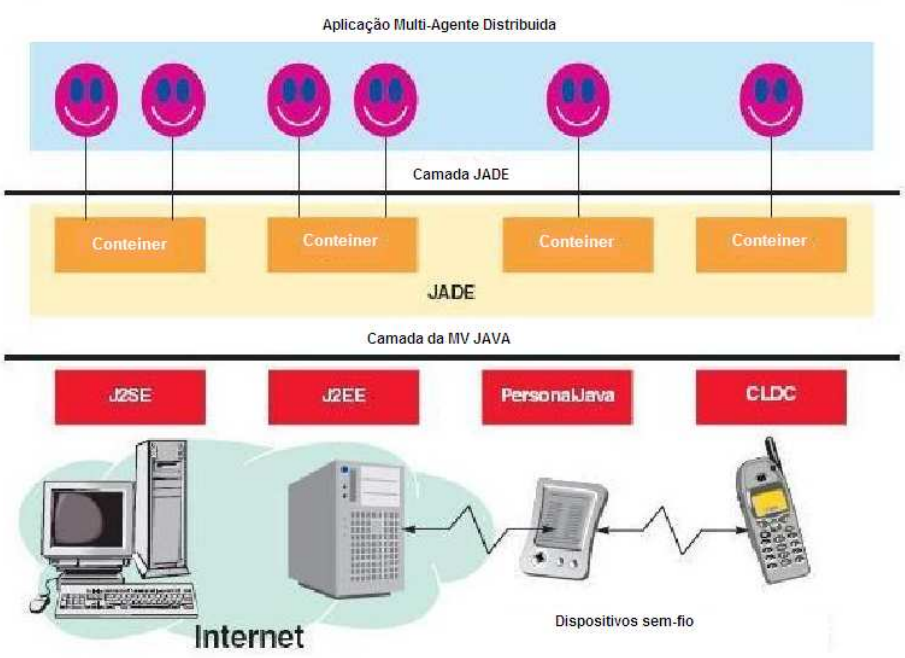

Figura 4.4: Arquitetura do *framework* JADE (9).

O *framework* pode ser observado a partir de duas visões: funcional e da aplicação. Do ponto de vista funcional, o JADE provê os serviços básicos necessários para aplicações *peer-to-peer* em ambientes *wired* ou *wireless*. O JADE permite que cada agente, dentro do paradigma *peer-to-peer*, descubra dinamicamente os outros agentes, que fazem parte do sitema multiagente, e

[7]Como o dispositivo pode ser wireless com J2ME ou wired executado em *desktop* com J2Se ou J2EE

estabeleça comunicação entre eles (9). Do ponto de vista da aplicação, cada agente é identificado por um nome IDA ou(identificador único) e provê um conjunto de serviços. O agente pode registrar e modificar seus serviços e/ou buscar por agentes, dentro do SMA, que forneçam um determinado serviço, pode controlar seu ciclo de vida e, em particular, estabelecer comunicação com todos os outros *peers* (topologia *peer-to-peer*).

A comunicação estabelecida entre os agentes, por troca de mensagens, pode ser feita de maneira assíncrona. Segundo Bellifemine (9), este é um modelo de comunicação usado para comunicações entre entidades heterogêneas que não sabem nada sobre as demais entidades com as quais estabelecem comunicação. A estrutura da mensagem está de acordo com a linguagem ACL, definida pela FIPA (36).

Podemos citar outras ferramentas importantes pesquisadas e utilizadas neste trabalho:

- O Biojava (bio) - é um projeto de código aberto que visa disponibilizar ferramentas, desenvolvidas em Java, para processamento de dados biológicos. Estas ferramentas incluem objetos para manipulação de seqüências, analisadores de arquivos, programação dinâmica, entre outros. Foram utilizadas algumas bibliotecas adaptadas para o parser explanadas na seção 6.2.

- JESS - Java Expert System Shell (jes) é um motor de inferência para construção de bases de conhecimento e inferência dirigida por padrão. Foi desenvolvido por Friedman-Hill no Sandia National Laboratories. Como o próprio nome sugere, o JESS é feito para interagir com Java, o que permite a criação de *software* Java com a capacidade de reagir usando conhecimento vindo das regras declarativas implementadas no JESS. O JESS é considerado leve e rápido sendo projetado para dar acesso a toda **API** Java. Desta forma, é possível criar objetos Java, invocar métodos dessa linguagem e implementar interfaces Java sem compilar nenhum código Java. O JESS foi utilizado neste trabalho para tratar inferências do agentes sendo integrado ao JADE.

- Protégé - com relação ao apoio ferramental para definição de ontologias, pode-se utilizar a ferramenta Protégé (pro) de maneira muito prática. Protégé foi desenvolvida como uma ferramenta de código livre, permitindo assim que possa ser realizada sua customização de

modo a compatibilizá-la com outras ferramentas para criação de ontologias, bem como permite a geração automática de código. Isso é possível porque podemos definir em Protégé uma ontologia em um formato abstrato, que pode então ser convertido para linguagens específicas, através de *plug-ins*. Atualmente, Protégé é a única ferramenta através da qual se pode gerar o código de ontologias em JADE automaticamente, usando como formato abstrato na ferramenta o modelo de ontologias de JADE, sendo utilizado neste trabalho para tratar a ontologia do BLAST e FASTA.

Capítulo 5

Um SMA para Anotação Manual

O objetivo do sistema é simular a análise que um biólogo faria para realizar uma anotação manual em projetos de sequenciamento de genomas. Para isso, o biólogo analisa as saídas de programas que computam similaridades ou outras funções celulares, relacionando estes resultados de acordo com seu conhecimento biológico, para concluir então qual a função do gene seqüenciado.

5.1 Arquitetura

A arquitetura de SMA proposta e utilizada uma plataforma comum e distribuída onde as tarefas são descentralizadas possibilitando criar um sistema mais flexível, com tempo de resposta adequado ao usuário e principalmente com resultados expressivos para a análise dos biólogos. A Figura 5.1 mostra a arquitetura proposta (iwg). Podemos observar que trata-se de uma arquitetura dividia em três camadas:

- Camada de Apresentação: responsável por coletar os parâmetros de entrada para o sistema e retornar o resultado do processamento. A entrada consiste na seqüência a ser analisada, os programas e os bancos que serão executados, ou a localização dos arquivos de saída desses programas. Estas funcionalidades são implementadas por um agente especializado, o *Agente de Interface*, que se encarrega de receber os parâmetros de entrada fazendo as requisições necessárias ao agente da camada imediatamente inferior e, ao receber as repostas desse agente, repassar os resultados.

- A Camada Colaborativa: responsável pela consolidação de todos os re-

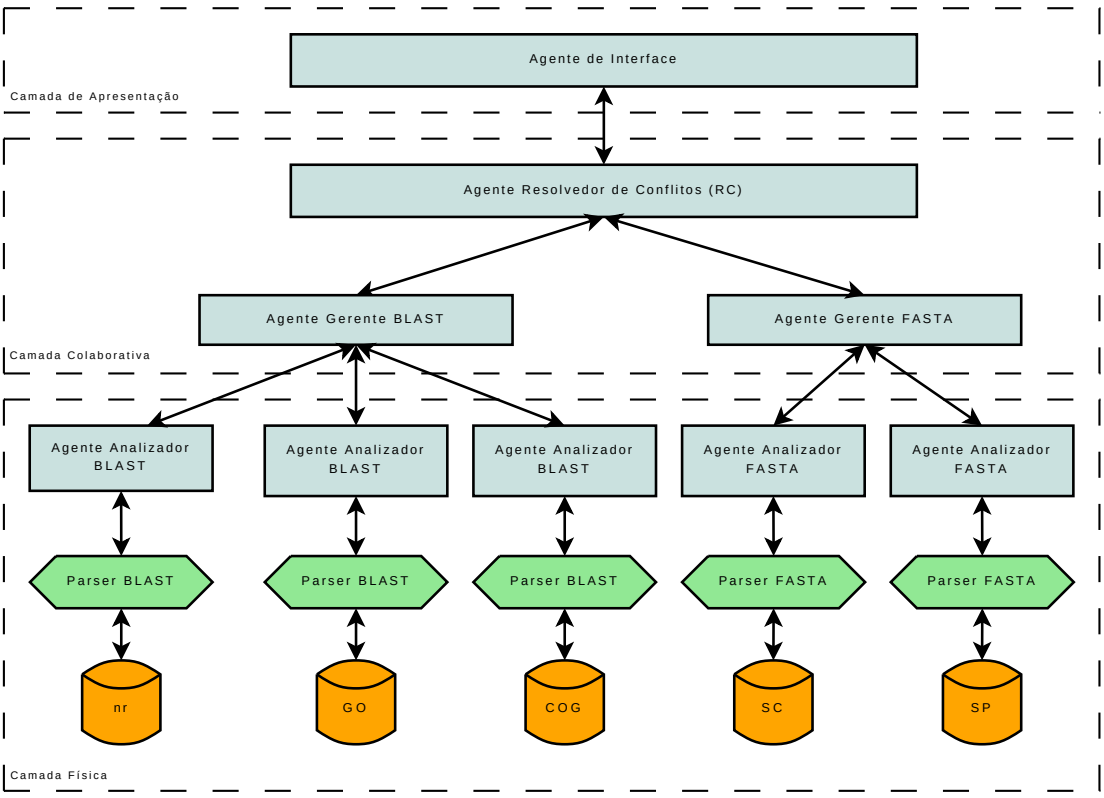

Figura 5.1: Proposta da arquitetura do SMA para anotação manual (iwg)

sultados provenientes da *Camada Física* e retorná-los para a *Camada de Apresentação*. Essa camada é composta pelos seguintes agentes:

- Agente Resolvedor de Conflitos (RC): recebe uma mensagem descrevendo uma configuração de serviço da *Camada de Apresentação*. Com isso o agente *RC* determina, de acordo com as especialidades dos *Agentes Gerentes*, quais deles devem ser solicitados para atender a solicitação. Essas solicitações são mensagens, requerindo sugestões, agrupadas de acordo com os programas aos quais os *Agentes Gerentes* são especializados. Após essas requisições o agente RC aguarda as respostas dos agentes para uma análise dos resultados retornados e decidirão sobre qual sugestão é a mais apropriada para ser enviada a *Camada de Apresentação*.
- Agente Gerente: recebe uma mensagem solicitando sugestões do agente *RC* conforme sua especialidade. De posse dessa solicitação o *Agente Gerente* verifica a quantidade de Bases de Dados e saídas de programas comparadores de seqüências a serem analisadas e de acordo com essa quantidade, o Agente Gerente solicita

Agentes Analisadores, alocando-os caso estejam disponíveis, para fazer a análise individual dessas saídas e Bases de Dados. Assim o *Agente Gerente* aguarda por todas as sugestões e as consolida utilizando suas regras internas. Cada *Agente Gerente* é especializado em um programa, por exemplo, um *Agente Gerente Blast* é especializado em avaliar e consolidar os resultados retornados por cada agente responsável por analisar isoladamente o BLAST com um banco em específico.

- Camada Física: responsável pelo acesso físicos aos arquivos e Bancos de Dados e análise individual deles. Essa camada é composta por vários *Agentes Analizadores* especializados em um programa e alocados para processar um determinado arquivo ou Base de Dados. Esses agentes recebem uma mensagem requisitando uma sugestão de um *Agente Gerente*. O *Agente Analizador*, conhecendo somente a Base de Dados, executa o programa para computar a similaridade da seqüencia assim gerando o arquivo de saída o qual é necessário para a análise. O programa a ser executado é definido de acordo com a especialidade do agente. Caso a localização desse arquivo for passada como parâmetro para o agente, esse não executa o programa. Uma estrutura de dados relativa ao arquivo gerado é construída através de uma Classe Parser e essa estrutura é analisada utilizando as regras internas do agente. Após esse processamento a sugestão é retornada para o *Agente Gerente*. Os *Agentes Analizadores* não permanecem vivos em momentos que são desnecessários, isto é, eles são criados quando existe uma processamento e quando encerram seu processamento são finalizados.

Existe um agente que atua em mais de uma camada, este agente é chamado Agente Gerenciador de Carga e atua nas camadas Físicas e Colaborativa, ele é responsável por criar e destruir agentes de acordo com as necessidades do sistema, mais especificamente, quando um Agente Gerente necessita de Agentes Analisadores o Agente Gerenciador de Carga se encarrega de criá-los e quando os Agentes Analisadores não são mais necessários são destruídos pelo Agente Gerenciador de Carga.

5.2 Funcionalidades da Plataforma

A arquitetura apresentada conta com algumas funcionalidaes que não foram descritas até o momento. São três os principais tipos de funcionalidades criadas:

- Ontologias - As ontologias foram criadas para mapear objetos de dados para a comunicação entre agentes. Foi desenvolvida uma ontologia para mapear a saída do BLAST, e outra para a saída do FASTA além de uma ontologia geral de comunicação entre agentes;

- Base de Conhecimento - Para dotar os agentes com capacidade de inferência foi criada uma base de conhecimentos usando o JESS. Basicamente são 3 regras que compõem essa base de conhecimento, que estão descritas na seção 6.3;

- Agentes Auxiliares - Devido a necessidades de controle de carga e de gerência de ciclo de vida dos agentes, foi definido um agente capaz de criar e destruir outros agentes de acordo com as necessidades do sistema. Por exemplo, quando um agente gerente recebe uma solicitação de serviço este deverá criar agentes executores para realizar a tarefa, mas o agente responsável por criar o requerido agente é o Agente Gerenciador de Carga, quando algum agente não é mais útil ele é destruído pelo Agente Gerenciador de Carga.

Capítulo 6

Aspectos da Implementacionais e Discussão dos Resultados Obtidos

O protótipo do SMA para anotação manual foi implementado utilizando o BLAST e o FASTA como programas comparadores de seqüências. Os bancos utilizados pelo BLAST foram o COG, GO e NR, enquanto o FASTA foi executado usando os bancos *Saccharomyces cereviseae*(sc) e *Schizosaccharomyces pombe*(sp). Podemos definir o pro cesso de implementação do SMA em cinco grandes etapas a seguir:

6.1 Ontologias

Foram usadas três ontologias conceitualmente agrupadas em uma ontologia global utilizada no projeto a saber: uma para especificar a comunicação entres agentes inteligentes e as outras duas para especificar o conhecimento relativo à análise feita pelos programas comparadores de seqüências (BLAST e FASTA). Para auxiliar a tarefa de desenvolvimento foi usado o Protégé para a criação de tais ontologias. A Figura 6.1 ilustra parte da implementação da ontologia BLAST utilizando o Protégé.

6.2 *Parsers*

Uma etapa fundamental da implementação do protótipo consiste em extrair corretamente todos os dados necessários dos arquivos de saída dos programas requeridos e mapear tais dados em uma estrutura de dados que possa ser compreendida pelos agentes. A parte de extração de dados

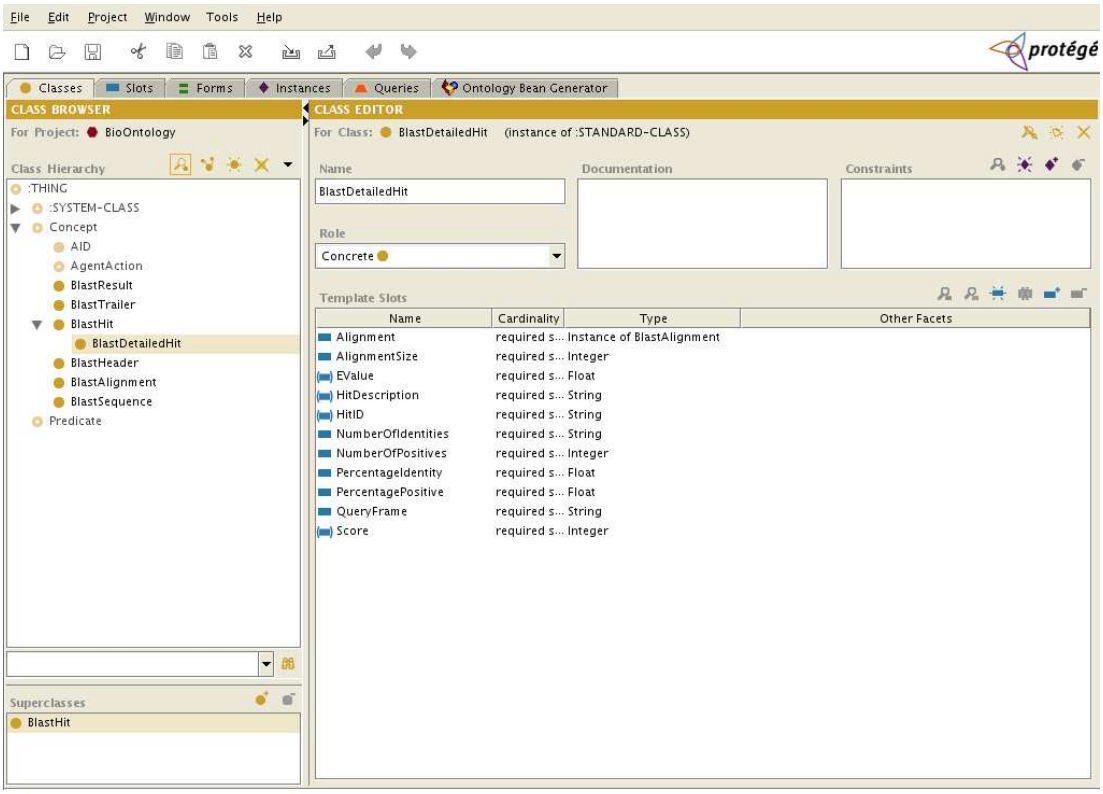

Figura 6.1: Parte da implementação da ontologia do BLAST no Protégé

foi adaptado de algumas bibliotecas do BioJava (bio). Foram criados dois Parsers: um para manipular arquivos de saída do BLAST e outro para manipular os arquivos de saída do FASTA. As classes que implementam estes parsers são:

- BlastParser - responsável por interpretar o arquivo de saída de uma execução do BLAST e transformá-la em uma estrutura de dados equivalente a ontologia BLAST para que um *Agente Analisador BLAST* possa analisá-la.
- FastaParser - analogamente ao BlastParser, este parser é responsável por interpretar o arquivo de saída de uma execução FASTA e transformá-la em uma estrutura de dados equivalente a ontologia FASTA para que um *Agente Analisador FASTA* possa analisá-la.

6.3 Bases de Conhecimentos

A base de conhecimento, que os agentes da *Camada Colaborativa e Física* usam, é a essência de inferência de todo o projeto, pois essa base

contém a inteligência dos agentes. Esta inteligência, em uma perspectiva geral, permite julgar e determinar qual deve ser a sugestão que o sistema retornará ao biólogo. Para entendermos como avaliar as saídas dos programas comparadores de seqüências e definirmos quais as regras a serem utilizadas para definirmos a sugestão a informar, foi necessário trabalhar em conjunto com biólogos especializados em anotação manual.

Na implementação desta base de conhecimentos foi utilizado a linguagem JESS, que possui um suporte adequado para tal função, segundo as necessidades do projeto. Existem três regras implementadas na base de conhecimento para agentes especializados em BLAST ou FASTA, sendo que as três regras são utilizados pelos *Agentes Gerentes* e também pelos *Agentes Analisadores*. As regras têm por objetivo selecionar uma sugestão mais apropriada dentre um conjunto de alinhamentos gerados nas saídas BLAST e FASTA. Essa regras atuam sobre os valores *e-value* e *score* de cada alinhamento e são as seguintes, tanto para os agentes FASTA como para os agentes BLAST:

- Verificar a existência de alinhamentos cujo o *e-value* seja maior ou igual a 10^{-5}.
- Dentre os alinhamentos que atendem a restrição anterior selecionar o menor *e-value*.
- Caso existem dois *e-values* iguais, selecionar o de maior *score*.

O código destas regras estão dispostos no Apêndice A.

6.4 Comunicação entre os agentes

A plataforma JADE utilizada neste trabalho implementa a maioria dos protocolos definidos pela FIPA. Neste trabalho foi utilizado o protocolo Responder (9), é um protocolo relativamente simples e atende a todas as necessidades de comunição dos agentes envolvidos. Este protocolo se comporta de acordo com o diagrama da Figura 6.2. A idéia do protocolo consiste em, ao perceber um ato de comunicação, o agente pode se comportar da seguinte maneira:

- O agente sinaliza que não entende a mensagem;

- O agente recusa a mensagem e informa o motivo de tal recusa e
- O agente aceita a mensagem, e quando isso ocorre existem outros três comportamentos possíveis:
  - O agente falha ao tentar processar o conteúdo da mensagem;
  - O agente consegue processar a mensagem e termina a ação e retornando uma e informação
  - O agente consegue processar a mensagem e caso o ato de comunicação tenha sido uma *query* retorna como resultado o processamento dessa *query*

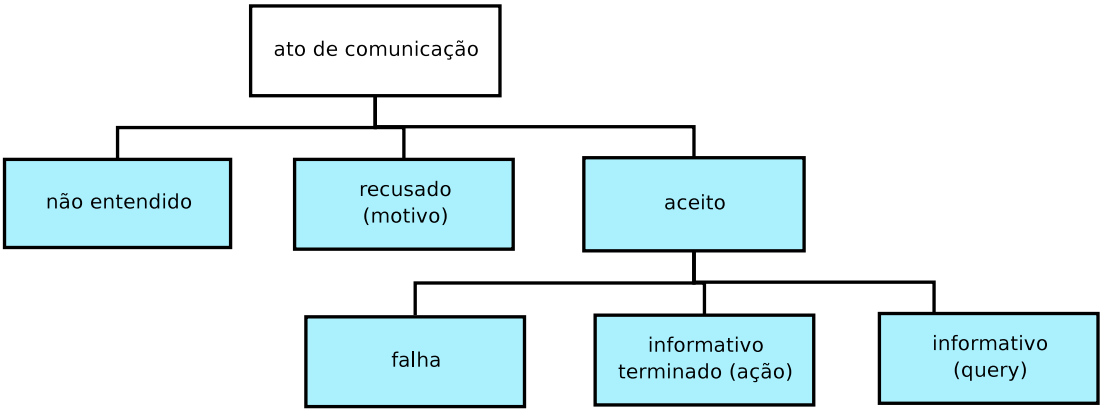

Figura 6.2: Diagrama do protocolo de comunicação Responder (9)

6.5 Agentes

Nesta seção caracterizaremos alguns detalhes da implementação dos agentes envolvidos neste trabalho.

Os agentes são classes especializadas da classe *Agent*, que o *framework JADE* disponibiliza para implementação de agentes. Essa classe implementa métodos de envio e recebimento de mensagens e todos outros métodos necessários para o bom funcionamento do agente dentro da plataforma. Desta forma é possível definir comportamentos dos agentes (comportamentos são as ações que um agente deve tomar). Esses comportamentos foram implementados especializando duas classes do *framework JADE*, a *AchieveREInitiator* e a *AchieveREResponder*, as quais são implementações de Máquinas de Estados que definem em conjunto o protocolo de comunicação Re-

sponder. *AchieveREInitiator* é a classe que inicia a comunicação e a classe *AchieveREResponder* responde a comunicação iniciada.

Existe um agente que não foi apresentado na arquitetura do trabalho, por ser um agente auxiliar a toda a arquitetura. Este agente é responsável gerenciar o ciclo de vida e alocação dos agentes da Camada Física desta arquitetura e de fazer o controle de carga do sistema, matando agentes que não estão sendo usados ou inicializando agentes que serão usados. Quando é dito que um *Agente Gerente* solicita e aloca caso disponível outro agente, na verdade eles o fazem de maneira indireta pois quem cria e aloca os agentes dessa camada é este agente chamado de *Agente Gerenciador de Carga*.

O *Agente de Interface* possui um único comportamento, o qual é responsável por iniciar uma comunicação. Esse é reponsável por receber os parâmetros que configuram as características desejadas para a anotação, através de uma interface, e repassar tais configurações adequadamente para o *Agente Resolvedor de Conflitos*. Feito isso ele espera pelo resultado desse ato de comunicação. Ao receber a resposta, ele a repassa através de sua interface.

O *Agente Resolvedor de Conflitos* possui dois comportamentos que implementam dois protocolos *Responder* aninhados. A instância do protocolo mais externa responde a comunicação vinda do *Agente de Interface* e quando recebida essa mensagem a segunda instância do protocolo inicia uma comunicação com os *Agentes Gerentes*. Iniciada essa comunicação o *Agente Resolvedor de Conflitos* espera por uma resposta e somente após o recebimento de todas as respostas ele consolida todos os resultados obtidos e da continuidade a comunicação com o *Agente de Interface*.

O *Agente Gerente* possui três comportamentos que implementam protocolos *Responder* aninhados, um externo e dois internos a esse. O comportamento mais externo recebe uma mensagem do *Agente Resolvedor de Conflitos* e verifica quantos *Agentes Analizadores* serão necessários para o processamento. Com isso, o primeiro comportamento interno inicia uma comunicação com o *Agente Gerenciador de Recursos* e solicita uma quantidade de *Agentes Analizadores*. Ao receber a resposta, o *Agente Gerente* verifica se foi possível alocar *Agentes Analizadores*. Se a alocação não foi feita, então continua-se o processo de comunicação com o *Agente Resolvedor de Conflitos* informando uma falha no processamento. Porém se o *Agente Gerenciador*

de Recursos alocou agentes para esse *Agente Gerente*, o segundo comportamento interno inicia uma comunicação com cada agente alocado solicitando uma sugestão por Base de Dados ou arquivo de resultados. Ao receber essa resposta, ele analisará todas as sugestões geradas e retomando a comunicação com o *Agente Resolvedor de Conflitos* envia-se a sugestão mais apropriada.

O *Agente Analizador* possui somente um comportamento implementando a resposta a comunicação iniciada por um *Agente Gerente*. Com o recebimento de uma requisição, o *Agente Analizador* instancia uma *Parser* apropriado para gerar a estrutura de dados a ser analisada. Após essa análise é enviada uma responta ao *Agente Gerente* contendo uma sugestão.

O cliclo de vida dos *Agente Analizador* incia-se quando existe uma solicitação de um agente desse tipo e termina quando esse agente não esta mais alocado para nenhum *Agente Gerente*. Esse controle todo é feito pelo *Agente Gerenciador de Recursos*.

6.6 Discussão

O protótipo implementado foi validado em um projeto de sequenciamento de genoma do laboratório de Bioinformática do Instituo de Biologia do UnB: Projeto Genoma Funcional do Paracoccidioides brasiliensis. Passaremos a descrever os resultados alcançados neste trabalho.

Foi implemetado um protótipo de SMA conforme a arquitetura projetada capaz de suportar as características do SMA e capaz de suportar também o acréscimo de novos agentes, caso necessário.O protótipo foi implementado incluindo:

- Todos os agentes descritos na arquitetura proposta além do Agente Gerenciador de Carga;
- Ontologia geral de comunicação baseadas no BLAST e no FASTA e foi criada uma ontologia geral de comunicação;
- Regras de inferência implementadas no JESS que compõem a Base de Conhecimento dos agentes do SMA;
- Parsers para as saídas dos programas BLAST e FASTA.

Realizou-se um estudo de caso utilizando o SMA desenvolvido. Foi utilizado o projeto genoma pb para comparar os resultados obtidos com o SMA e o que foi anotado manualmente pelos biólogos. O estudo de caso consistiu em executar o protótipo que criamos com a base de dados do Projeto Genoma Pb para gerar sugestões de anotação e comparar estas sugestões com o que os biólogos anotaram efetivamente no projeto. A partir dos resultados obtidos foi gerada a tabela 6.1 que mostra os resultados alcançados pelo protótipo.

Número total de genes	6.107
Número de genes anotados	3.774
Número de anotações sugeridas	3.502
Número de acertos do SMA	1.547 (44, 1%)
Número de sugestões para genes não anotados	336

Tabela 6.1: Tabela de resultados obtidos pelo SMA comparados com o Projeto Genoma Pb

O SMA sugeriu 3.502 anotações e acertou 1.547 quando comparado a anotação manual feita pelos biólogos, ou seja 44,1 % de acertos. Além disso o sistema sugeriu 336 anotações para os 2.333 genes não anotados. Este índice foi alcançado utilizando apenas as três regras descritas na Seção 6.3. O tempo de processamento necessário para fazer as sugestões de todo o Projeto Genoma Pb, 6.107 seqüências, foi de aproximadamente 1 hora e 30 minutos, tempo curto quando comparado a quantidade de seqüências analisadas.

Este resultados são consideravelmente bons e podem ser melhorados acrescentando regras mais específicas ou mais agentes específicos para outros programas, melhorando a resolução de conflitos.

Capítulo 7

Conclusões e Trabalhos Futuros

Neste trabalho, implementamos um protótipo de SMA conforme arquitetura proposta para auxiliar a fase de anotação manual de projetos de seqüenciamento de genomas. Nos testes realizados, o sistema acertou aproximadamente 44% das suas sugestões de anotações manuais de acordo com as realizadas pelos biólogos no Projeto Genoma *Paracoccidioides brasiliensis*, e além disso o sistema sugeriu 336 anotações para os 2.333 gentes não anotados.

Pesquisas interessantes podem seguir este trabalho:

- Distribuir a carga de processamento utilizada pelo sistema de anotação manual, diminuindo o tempo de processamento e aproveitando recursos de processamento distribuído e paralelo através do uso de protocolos como o contract net.

- Integrar o sistema de anotação manual com o *framework* Timina (tim), tornando a ferramenta de anotação manual um *plugin* para o Timina, dispensando desta forma a necessidade de adaptações do SMA ao projeto de seqüenciamento implantado.

- Aumentar o número de regras utilizadas pelos *Agentes Gerentes* e pelos *Agentes Analisadores* para especializar a função de determinação de genes, elevando o percentual de acerto obtido pelo SMA.

- Adicionar novas regras ao *Agente Resolvedor de Conflitos* visando melhorar a escolha e consolidação de resultados gerados pelos *Agentes Gerentes* e conseqüentemente obter um melhor percentual de acerto obtidos pelo SMA.

- Adicionar outros programas e desenvolver outros ao *Agentes Gerentes*, *Agentes Analisadores* o que levará também a um aumento no percentual de acertos do SMA.

- Adicionar ferramentas de mineração de dados e análises ontológicas aos *Agentes Gerentes* e ao *Agente Resolvedor de Conflitos* o que também aumentará o percentual de acertos do SMA.

- Trabalhar no sentido de alterar a abordagem reativa dos agentes para uma abordagem orientada a objetos possivelmente alterando ou complementando mecanismos de inferência de regras com escalonamento regressivo além do progressivo utilizado.

- Aplicar em outros projetos de sequenciamento de genomas já anotados e até em projetos de genomas nunca anotados.

- Implementar o protótipo de forma distribuída melhorando a utilização de recursos.

Em fim, este trabalho possibilita muitos trabalhos futuros o que realmente comprova sua importância na área de Bioinformática.

Apêndice A

Código JESS

```
(defmodule Evalue)

(defrule Exists_Evalue_Above_Limit
"Activate EvalueAnalysis module if there is at
 least one evalue smaller than ?*maxEvalue*"
(exists (BlastHit
(hitEvalue ?evalue&:(<= ?evalue ?*maxEvalue*))
  ))
  =>
  (focus EvalueAnalysis)
  (run)
)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;  Module EvalueAnalysis ;;;;

(defmodule EvalueAnalysis)

(defrule Assert_Sugestion_Facts
  "Asserts Sugestion Facts for each BlastHit"
  (logical (BlastHit(OBJECT ?blastHit)
                    (hitID ?id)
                    (hitEvalue ?evalue&:
                               (<= ?evalue ?*maxEvalue*))
                    (hitScore ?score))
 (MAIN::Database ?db)
```

```
   (MAIN::Algorithm ?al)
   )
  =>
(bind ?sug (new ontology.Sugestion))
  (?sug setHit ?blastHit)
  (?sug addExplanation
    (str-cat "E-value menor que " ?*maxEvalue*)
   )
  (?sug setDataBase ?db)
  (?sug setAlgorithm ?al)
(add ?sug)
 )

(defrule Best_Evalue
  "Retract sugestion that will be not used"
  (Sugestion (hit ?hit1)(OBJECT ?sug))
  ?retract <- (Sugestion (hit ?hit2))
  (and (test (>= (?hit2 getHitEvalue)(?hit1 getHitEvalue)))
       (test (< (?hit2 getHitScore)(?hit1 getHitScore)))
  )
  =>
(retract ?retract)
  (?sug addExplanation "Menor Evalue com Maior Score")
  (update ?sug)
)
```

Apêndice B

ENIA 2007

O artigo a seguir foi escrito com base neste trabalho e em seus resultados e foi publicado nos anais do Encontro Nacional de Inteligência Artificial do Ano de 2007 (ENIA'07).

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

Richardson Silva Lima[1], Célia Ghedini Ralha[1], Maria Emília Machado T.Walter[1], HugoWruck Schneider[1], Anderson Gray F. Pereira[1], Marcelo Macedo Brígido[2]

Departamento de Ciência da Computação, Instituto de Cências Exatas

[1]Universidade de Brasília, Campus Universitário Darcy Ribeiro Caixa Postal 4466, Brasília-Brasil, CEP 70.910-900

[2]Instituto de Biologia, Universidade de Brasília Campus Universitário Darcy Ribeiro, Brasília-Brasil, CEP 70.910-900

{rlima,ghedini}@cic.unb.br, {mariaemilia,brigido}@unb.br

{0332658,0336416}@aluno.unb.br

Abstract *Genome sequencing projects identify biological sequences of organisms and their functions. The discovery of biological functions constitutes the annotation phase, that is divided into automatic and manual. Automatic annotation has the objective of inferring functions to the project sequences, using databases containing biological sequences and previously determined functions. Manual annotation is done by biologists, that decide the functions using their biological knowledge. This work presents BioAgents, a system that uses the Multiagent paradigm to support manual annotation.*

BioAgents provides interaction of different agents using the automatic annotation outputs, and suggests manual annotations that must be validated by biologists.

Resumo *Projetos de seqüenciamento de genomas identificam seqüências biológicas de organismos e suas funções. A descoberta das funções biológicas constitui a fase de anotação, dividida em automática e manual. A anotação automática visa atribuir funções às seqüências do projeto, usando bancos de dados de seqüências biológicas com funções previamente determinadas. A anotação manual é feita por biólogos, que decidem as funções usando seu conhecimento biológico. Este trabalho apresenta o BioAgents, um sistema que utiliza o paradigma Multiagente para apoiar a anotação manual. BioAgents provê a interação entre diferentes agentes usando os resultados da anotação automática, e sugere anotações manuais que deverão ser validadas pelos biólogos.*

1. Introdução

Em 1953, Watson e Crick propuseram uma estrutura molecular para o [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].

[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.

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 (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:

- 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 resulta-

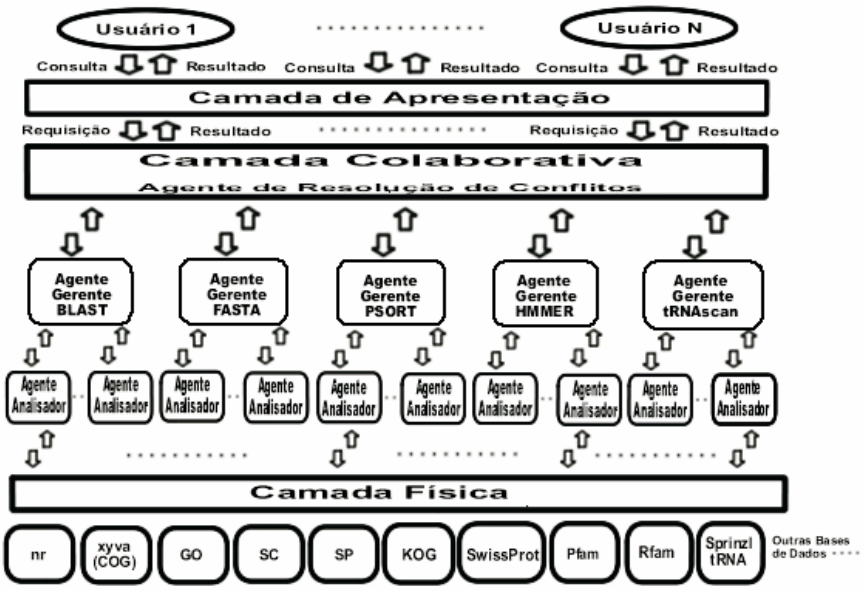

Figura B.1: A arquitetura em três camadas do sistema *BioAgents*.

dos 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 cerevisiae* (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.

[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`).

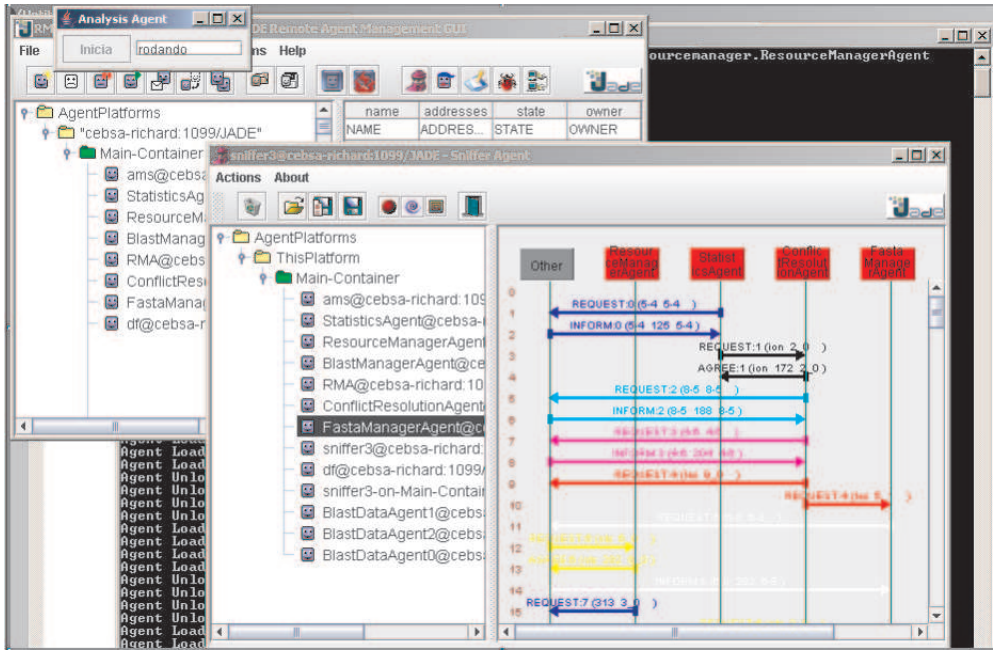

Figura B.2: *Screenshot* da tela de execução e do *sniffer* dos agentes do *BioAgents* no framework *JADE*.

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 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.

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

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, 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 incluindo 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 403-410.

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:D16-D20.

Decker, K., Zheng, X., and Schmidt, C. (2001). A multi-agent system for automated genomic annotation. In AGENTS 01: Proceedings of the 5th 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):2503-2509.

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, 409:860-921.

Lemos, M. (2004). Workflow para bioinform´atica. PhD thesis, Pontif´ýcia 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 IGWD 05 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 and Sons, LTD, England.

Apêndice C

IWGD 2007

O artigo a seguir foi escrito com base neste trabalho e em seus resultados e foi aceito no International Workshop on Genomic Databases do ano de 2007 (IWGD'07).

BioAgents: A Multiagent System for Manual Annotation on Genome Sequencing Projects

Richardson S. Lima[1], Célia G. Ralha[1], Maria Emilia M. T. Walter[1], Hugo W. Schneider[1], Anderson G. F. Pereira[1] and Marcelo M. Brígido[2]

[1]Computer Science Department, [2]Biology Institute University of Brasilia

Campus Universitário Darcy Ribeiro Caixa Postal 4466 - CEP 70.919-970

Brasília, Brazil Email address: 1rlima, ghedini@cic.unb.br, 1,2mariaemilia, brigido@unb.br, 10332658, 0336416@aluno.unb.br

On genome sequencing projects, the annotation phase has the objective to identify biological functions to the DNA sequences obtained on the project. The annotation phase is divided on two tasks. Automatic annotation proposes biological functions to each sequence, based on approximated comparison algorithms and databases containing sequences with corresponding functions. Manual annotation guarantees accuracy and correctness to each sequence function. So, assisting the biologists in this task with computational tools will certainly improve the final annotation. To do this, we considered as hypothesis the fact that the biologists decide - based in the tacit knowledge, acquired with professional experience - the annotation of any sequence. In this context, this work presents a multiagent system to help the biologists to do the manual annotation on genome sequencing projects. We developed a system using a multiagent architecture, according to a blackboard approach (Figure 1). The architecture is divided

into three layers: interface, collaborative and physical layer. The interface layer receives the requests and returns the results to the user. The collaborative layer, the architecture core, has agents that interact with manager agents (specialized in BLAST and FASTA), local databases and different knowledge sources to suggest annotations to be sent for 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 data databases and specific knowledge sources, based in production rules, to suggest annotations that must be later validated by the biologists.

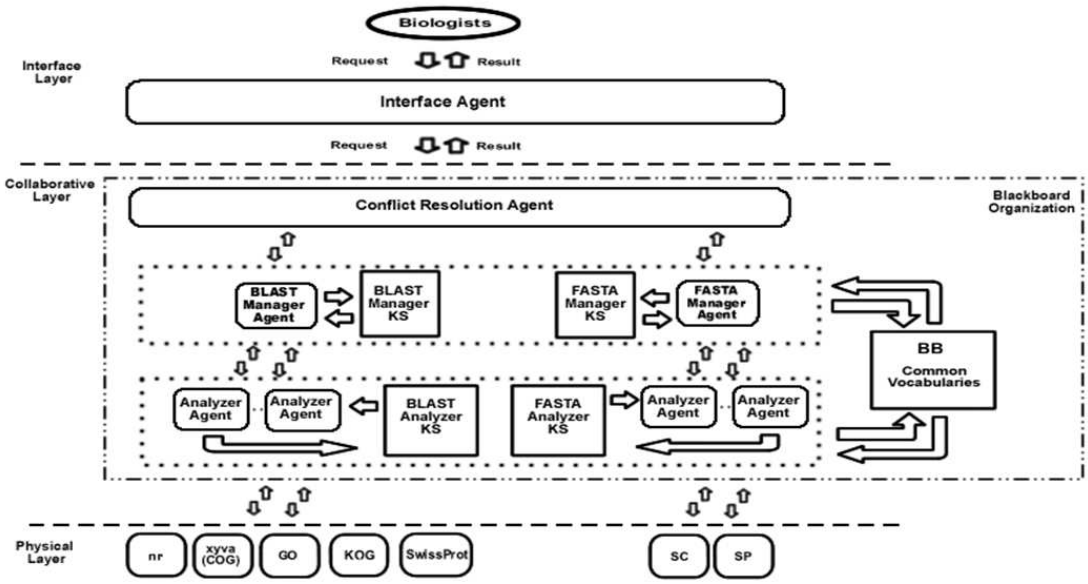

Figura C.1: BioAgents Architecture

Figure1-BioAgents Architecture. The system was developed using the JADE framework. To validate our system, we used data from the manual annotations of the Paraccocidiodes brasiliensis Genome Project (Genome Pb Project) and the Paullinia Cupana Genome Project (Genome Guarana Project). The production rules, implemented in the system expertise module, used only the parameters: e-value less than or equal to 10-5 and score greater than or equal to 40%. For the Genome Pb Project, having 6,107 sequences with 2,327 manually annotated, our system suggested 3,496 annotations, where 1,547 (44.25%) were correct when compared to the previous manual annotation. For the Genome Guarana Project, having 8,597 sequences with 7,725 manually annotated, the system suggested 6,478 annotations, where 2.938 (45.35%) were correct. These experiments indicated

that our multiagent system can really help biologists during the manual annotation phase of a genome sequencing projects. We intend to improve the expertise of the system in order to obtain better results.

References

1. Ding Li, et al: **EAnnot: A genome annotation tool using experimental evidence**. Genome Research 2004, 14(12): 25032509.

2. Wooldridge M: An Introduction to MultiAgent Systems. John Wiley & Sons Ltd; 2002.

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

4. Decker K, et al: **BioMAS: A MultiAgent System for Genomic Annotation**. International Journal of Cooperative Information Systems 2002, 11(3): 265292.

5. Andrade MA, et al: **Automated genome sequence analysis and annotation**. Bioinformatics 1999, 15(5): 391412.

6. Bazzan ALC, et al: **ATUCG - An Agent-Based Environment for Automatic Annotation of Genomes**. International Journal of Cooperative Information Systems 2003, 12(2): 241273.

7. Nascimento LV, Bazzan ALC: **An agent-based system for re-annotation of genomes**. Genetics and Molecular Research 2005, 4(3): 571580.

8. **JADE-Java Agent Development Framework** `http://jade.tilab.com`].

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

10. Altschul SF, et al: **Basic Local Alignment Search Tool**. Journal of Molecular Biology 1990, 215(3): 403410.

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

12. **Genome Pb Project** `http://dna.biomol.unb.br/Pb/`

13. **Genome Guaraná Project** `http://dna.biomol.unb.br/GR/`

Referências Bibliográficas

[tig] TIGR – The Institute For Genome Research. http://www.tigr.org/.

[iwg] IWGD 2005. http://www.biowebdb.org/iwgd05/.

[san] Sanger Center. http://www.sanger.ac.uk/.

[doe] DOE Joint Genome Institute. http://www.jgi.doe.gov/.

[wus] Universidade de Washington em St. http://www.wustl.edu/.

[gol] GOLD – Genomes OnLine Database. http://wit.integratedgenomics.com/GOLD/.

[ons] ONSA – Organization for Nucleotide Sequencing and Analysis. http://watson.fapesp.br/onsa/Genoma3.htm.

[1] Genolyptus. http://ftp.mct.gov.br/especial/genolyptus3.htm.

[pb] Projeto Genoma Pb. http://www.biomol.unb.br/Pb.

[2] GenBank. http://www.ncbi.nlm.nih.gov/Genbank/.

[ncb] National Center for Biotechnology Information. http://www.ncbi.nlm.nih.gov/.

[pfa] Pfam. http://www.sanger.ac.uk/Software/Pfam/.

[pso] PSORT. http://www.psort.org/.

[go] Gene Ontology. http://www.geneontology.org/.

[bio] BioJava. http://biojava.org/.

[jes] JESS. http://www.jessrules.com/.

[pro] Protégé. http://protege.stanford.edu/.

[tim] Timina. http://sourceforge.net/projects/timina/.

[3] Adams, M. D., Kelley, J. M., Gocayne, J. D., Dubnick, M., Polmeropolus, M. H., Xiao, H., Merril, C. R., Wu, A., Olde, B., Moreno, R. F., Kerlavage, A. R., Mccombie, W. R., & Venter, J. C. (1991). Complementary DNA sequencing: expressed sequence tags and human genome project. *Science*, 252, 1651–1656.

[4] Alberts, B., Bray, D., Johnson, Lewis, J., Raff, M., Roberts, K., & Walter, P. (1999). *Fundamentos da Biologia Celular*. Porto Alegre, RS, Brasil: Artmed.

[5] Almeida, M. B. & Bax, M. P. (2003). Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção. *Revista Ciência da Informação*, 32(3).

[6] Altschul, S. F., Schaffer, T. L. M. A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17), 3389–3402.

[7] Andrade, R. V. (2002). Caracterização parcial do transcriptoma do fungo dimórfico e patogênico *Paracoccidioides brasiliensis*. Master's thesis, Universidade de Brasília, Departamento de Biologia Molecular.

[8] Baker, P. G., Brass, A., Bechhofer, S., Goble, C., Paton, N., & Stevens, R. (1998). TAMBIS: Transparent access to multiple bioinformatics information sources. In J. Glasgow, T. Littlejohn, F. Major, R. Lathrop, D. Sankoff, & C. Sensen (Eds.), *6th Int. Conf. on Intelligent Systems for Molecular Biology* (pp. 25–34). Montreal, Canada: AAAI Press, Menlo Park.

[9] Bellifemine, F., Caire, G., Poggi, A., & Rimassa, G. (2003). *JADE - A White Paper*. White Paper Volume 3-Number 3-2003, TILAB - Telecom Italia Lab. http://jade.tilab.com/papers/2003/WhitePaperJADEEXP.pdf.

[10] Departamento de Química – UFSC (2002). O mundo das proteínas. *Revista Eletrônica do Departamento de Química*, (53).

[11] Eddy, S. (2002). A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an rna secondary structure. http://citeseer.ist.psu.edu/eddy02memoryefficient.html.

[12] Eddy, S. (2005). *INFERNAL User's Guide: Sequence analysis using profiles of RNA secondary structure consensus*. User guide, Howard Hughes Medical Institute and Dept. of Genetics. Saint Louis, USA. http://www.csc.fi/molbio/progs/infernal/Userguide.pdf.

[13] Eddy, S. R. (1999). Noncoding RNA genes. *Current Opinion in Genetics & Development*, 9, 695–699.

[14] Ferber, J. (1999). *Multi-Agent Systems*. England: Addison-Wesley.

[15] Gerhard, W. (2000). *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence*. MIT Press, second edition edition.

[16] Gibas, C. & Jambeck, P. (2001). *Developing Bioinformatics Computer Skills*. Sebastopol, CA, United States: O'Reilly.

[17] Gomez, A. P. (1998). *Knowledge Sharing and Reuse*. in The Hand Book of Apllied Expert Systems. CRC Press LLC: Boca Raton, FL.

[18] Green, E. D. (2001). Strategies for the systematic sequencing of complex genomes. *Nature Reviews Genetics*, 2(8), 573–583.

[19] Gruber, T. R. (1993). Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In N. Guarino & R. Poli (Eds.), *Formal Ontology in Conceptual Analysis and Knowledge Representation* Deventer, The Netherlands: Kluwer Academic Publishers.

[20] Honavar, V., Silvescu, A., Reinoso-Castillo, J., Andoff, C., & Dobbs, D. (2001). Ontology driven information extraction and knowledge acquisition from heterogeneous, distributed biological data sources.

[21] Lander, E. S., Linton, L. M., Birren, B., et al. (2001). Initial sequencing and analysis of the human genome. *Nature*, 409, 860–921.

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

[23] Lewin, B. (2001). *Genes VII*. Porto Alegre, RS, Brasil: Artmed.

[24] Lewis, S., Ashburner, M., & Reese, M. G. (2000). Annotating eukaryote genomes. *Current Opinion in Structural Biology*, 10(3), 349–354.

[25] López, M. (1999). Overview of the methodologies for building ontologies. Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods (KRR5), Stockholm, Sweden, August 1999. http://citeseer.ist.psu.edu/lpez99overview.html.

[26] Luscombe, N. M., Greenbaum, D., & Gerstein, M. (2001). What is bioinformatics? an introduction and overview. In R. HAUX & C. KULIKOWSKI (Eds.), *Yearbook Of Medical Informatics* 2001 (pp. 83–100). Internation Medical Informatics Association.

[27] Nascimento, L. V. & Bazzan, A. L. C. (2004). An agent-based system for re-annotation of genomes. *Terceiro Workshop Brasileiro de Bioinformática*, 10(01). http://www.inf.ufrgs.br/ mas/masbio/papers/artigo.pdf.

[28] Pearson, W. R. & Lipman, D. (1988). Improved tools for biological sequence comparison. *Proceedings Of The National Academy Of Science USA*, 85, 2444–2448.

[29] Pertsemlidis, A. & Fodon, III, J. W. (2001). Having a BLAST with bioinformatics (and avoiding BLASTphemy). *Genome Biology*, 2(10), 1–10.

[30] Russell, S. J. & Norvig, P. (2002). *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition edition.

[31] Schulze-Kremer, S. (1997). Adding semantics to genome databases: Towards an ontology for molecular biology. In T. Gaasterland, P. Karp, K. Karplus, C. Ouzounis, C. Sander, & A. Valencia (Eds.), *5th Int. Conf. on Intelligent Systems for Molecular Biology* (pp. 272–275). Halkidiki, Greece: AIII Press, Menlo Park.

[32] Schulze-Kremer, S. (1998). Ontologies for molecular biology. In *3rd Pacific Symposium on Biocomputing* (pp. 705–716).

[33] Setúbal, J. & Meidanis, J. (1997). *Introduction to Computational Molecular Biology*. Pacific Grove, CA, United States: Brooks/Cole Publishing Company.

[34] Simpson, A. J. G., Reinach, F. C., Arruda, P., & Others (2000). The genome sequence of the plant pathogen *Xylella fastidiosa*. *Nature*, 406(6792), 151–157.

[35] Sycara, K., Decker, K., Pannu, A., Williamson, M., & Zeng, D. (1996). Distributed intelligent agents. *IEEE Expert: Intelligent Systems and Their Applications*, 26(11), 36–46. http://citeseer.ist.psu.edu/article/sycara96distributed.html.

[36] Team, D. (2006). Fipa web site. http://www.fipa.org/.

[37] Vasconcelos, A. T. R. (2003). The complete genome sequence of chromobacterium violaceum reveals remarkable and exploitable bacterial adaptability. *PNAS*, 1.

[38] Venter, J. C., Adams, M. D., Myers, E. W., & Others (2001). The sequence of the human genome. *Science*, 291, 1304–1351.

[39] Watson, J. D. & Crick, F. H. (1953). Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature*, 171, 737–738.

[40] Wolfberg, T. G. & Landsman, D. (1997). A comparison of expressed sequence tags (ESTs) to human genomic sequences. *Nucleic Acids Research*, 25(8), 1626–1632.

[41] Wolfberg, T. G. & Landsman, D. (2001). Expressed sequence tags (ESTs). In A. D. Baxevanis & B. F. F. Ouellette (Eds.), *Bioinformatics* chapter 12. New York, NY, United States: Wiley–Interscience, second edition.

[42] Wooldridge, M. (2002). *An Introduction to Multiagent Systems*. England: John Wiley & Sons LTD.