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

Uma Ferramenta Multiagente Baseada em Conhecimento para Anotação de Proteínas: um Estudo de Caso para o Fungo *Saccharomyces cerevisiae*

Daniel da Silva Souza

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Prof.ª Maria Emília Machado Telles Walter

Brasília
2014

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

Coordenadora: Prof.ª Dr.ª Alba Cristina Magalhães de Melo

Banca examinadora composta por:

Prof.ª Maria Emília Machado Telles Walter (Orientadora) — CIC/UnB
Prof.ª Dr.ª Natália Florencio Martins — Cenargen/Embrapa
Prof.ª Dr.ª Célia Ghedini Ralha — CIC/UnB

CIP — Catalogação Internacional na Publicação

da Silva Souza, Daniel.

Uma Ferramenta Multiagente Baseada em Conhecimento para Anotação de Proteínas: um Estudo de Caso para o Fungo *Saccharomyces cerevisiae* / Daniel da Silva Souza. Brasília : UnB, 2014.
78 p. : il. ; 29,5 cm.

Dissertação (Mestrado) — Universidade de Brasília, Brasília, 2014.

1. anotação de proteínas, 2. fungos, 3. sistemas multiagentes,
4. sistemas baseados em conhecimento.

CDU 004.4

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

Uma Ferramenta Multiagente Baseada em Conhecimento para Anotação de Proteínas: um Estudo de Caso para o Fungo *Saccharomyces cerevisiae*

Daniel da Silva Souza

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Prof.ª Maria Emília Machado Telles Walter (Orientadora)
CIC/UnB

Prof.ª Dr.ª Natália Florencio Martins
Cenargen/Embrapa

Prof.ª Dr.ª Célia Ghedini Ralha
CIC/UnB

Prof.ª Dr.ª Alba Cristina Magalhães de Melo
Coordenadora do Mestrado em Informática

Brasília, 15 de dezembro de 2014

Dedicatória

Dedico este trabalho às professoras Maria Emília e Célia Ralha, que tiveram um papel importantíssimo na minha vida, pois elas foram mais que professoras para mim. Graças a elas, decidi seguir a carreira acadêmica. Obrigado por tudo!

Agradecimentos

Agradeço à minha família, pela compreensão e apoio que me deram durante todo esse período.

À orientadora prof.ª Maria Emília, por ter acolhido e acreditado no projeto. Por todo apoio, confiança e paciência depositada em mim, desde à iniciação científica até o presente momento.

À prof.ª Célia Ralha, também pela sua confiança, pelos conselhos e direções ao longo de todos esses anos.

Aos pesquisadores da Embrapa Roberto Togawa, Natália Martins e Priscila Grynberg, pelas valiosas contribuições e que, desde sempre, confiaram e acompanharam o desenvolvimento deste trabalho.

À Tainá Raiol, também por suas valiosas contribuições, sendo uma delas, o ponto de engate que levou a concretização deste trabalho.

À meu amigo Marcius Marques, por toda ajuda e foco nos estudos durante esses anos.

Aos amigos do IFG Thiago Peixoto e Waldeyr Mendes, pela amizade e apoio durante o mestrado.

Ao meu amigo João Victor, que sempre esteve presente para fornecer algum apoio, não apenas nos estudos.

Ao CNPq, pelo apoio à pesquisa.

Resumo

Identificar funções biológicas das sequências é uma atividade chave em projetos genomas. Esta tarefa é realizada na etapa de anotação, que possui duas fases. Na fase manual, biólogos utilizam seu conhecimento e experiência determinar a função de cada sequência, baseada nos resultados produzidos pela fase automática, onde ferramentas e bancos de dados são utilizados para predizer uma anotação funcional. Esta dissertação propõe BioAgents-Prot, uma ferramenta multiagente baseada em conhecimento, que simula o conhecimento e experiência dos biólogos para anotação de proteínas. BioAgents-Prot foi definido com uma abordagem de agentes cooperativos, onde diferentes agentes especializados trabalham em conjunto na tentativa de sugerir uma anotação manual adequada. A arquitetura proposta em três camadas foi desenvolvida com *Java Agent DEvelopment Framework* - JADE e Drools, um motor de inferência baseado em regras. Para avaliar o desempenho do BioAgents-Prot, as anotações dos transcritos do fungo *Saccharomyces cerevisiae* foram comparadas com as anotações sugeridas pelo sistema. Usando regras básicas que representam o raciocínio de anotação, obtemos 95.84% de *sensibilidade*, 93.22% de *especificidade*, 98.40% de *F1-score* e 0.80 de *MCC*, que demonstram a utilidade do BioAgents-Prot na etapa de anotação em projetos transcritoma.

Palavras-chave: anotação de proteínas, fungos, sistemas multiagentes, sistemas baseados em conhecimento.

Abstract

Identifying biological function of sequences is a key activity in genome projects. This task is done in the annotation step, which has two phases. In the manual phase, biologists use their knowledge and experience to determine the function for each sequence, based on the results produced by the automatic phase, where tools and data bases are used to predict functional annotation. This dissertation presents *BioAgents-Prot*, a knowledge based multiagent tool, which simulates biologists expertise to annotate proteins. *BioAgents-Prot* is defined with an approach of cooperative agents, where specialized intelligent agents work together to suggest proper manual annotation. The proposed three-layer architecture was implemented with *Java Agent DEvelopment Framework-JADE* and Drools (a rule-based inference engine). To assess performance, transcript annotations of the *Saccharomyces cerevisiae* fungus were compared to the annotations suggested by BioAgents-Prot. Using basic rules that represents the annotation reasoning, we obtained 95.84% of *sensitivity*, 93.22% of *specificity*, 98.40% of *F1-score* and 0.80 of *MCC*, which shows the usefulness of BioAgents-Prot in annotation step of transcriptome projects.

Keywords: protein annotation, fungi, multiagent systems, knowledge-based systems.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O Projeto Genoma Humano [46, 82] foi resultado de um esforço colaborativo internacional para sequenciar e mapear o genoma humano, iniciado na década de 1990. No âmbito desse projeto, desenvolveram-se técnicas laboratoriais e computacionais para analisar as sequências de DNA[1] geradas em várias instituições em diferentes países. Neste contexto, surgiu a Bioinformática, uma área que visava inicialmente apenas dar suporte a projetos genoma, mas que hoje constitui-se em uma interessante área de pesquisa.

Recentemente, técnicas de sequenciamento em larga escala [39], usadas em projetos genoma e transcritoma em todo o mundo, vêm produzindo um enorme volume de sequências biológicas com funções ainda desconhecidas, gerando a necessidade de criar metodologias e ferramentas eficientes para dar suporte computacional a esses projetos de sequenciamento. Nesses projetos, biólogos visam identificar sequências de DNA e/ou RNA[2], assim como suas funções nos mecanismos celulares. Essas funções, juntamente com outras características biológicas, são determinadas na fase de anotação, em *workflows* de bioinformática, geralmente constituídos de quatro etapas:

- **sequenciamento**: Existem diversos métodos e sequenciadores, que são adotados de acordo com as necessidades de cada projeto. Os sequenciadores de alto desempenho [39] produzem milhões de pequenos fragmentos do DNA ou RNA em um curto espaço de tempo;
- **filtragem e controle de qualidade dos fragmentos**: Os fragmentos gerados podem ser filtrados, removendo-se (corte ou *trimming*) sequências ou bases de baixa qualidade, adaptadores e contaminantes, a fim de garantir um conjunto de fragmentos com qualidade mínima, conferindo confiabilidade aos outros passos da análise computacional. A filtragem é feita a partir de informações de características específicas de cada sequenciador;

[1] ácido desoxirribonucleico *(deoxyribonucleic acid - DNA)*
[2] ácido ribonucleico *(ribonucleic acid - RNA)*

- **montagem *de novo* e/ou mapeamento**: O processo de reconstrução do DNA ou RNA pode ocorrer por: montagem *de novo*, em que uma sequência consenso é montada a partir de um alinhamento múltiplo de fragmentos que apresentam superposição entre si; ou por mapeamento de fragmentos em um genoma de referência, em que centenas de fragmentos "são mapeados" em certas regiões em um genoma de referência. Várias ferramentas, com diferentes metodologias, foram propostas para ambas as estratégias, tendo sido dezenas dessas ferramentas analisadas e comparadas por Miller et al. [54];

- **anotação**: A anotação é realizada para atribuir funções às sequências identificadas do(s) organismo(s) em estudo. Devido ao enorme volume de dados não caracterizados gerado, a anotação requer automação, uma tarefa complexa que inclui diferentes metodologias e estratégias de anotação para predição de função das proteínas [61], e de RNAs não-codificadores (ncRNAs). Biólogos geralmente combinam essas metodologias e estratégias em *pipelines* de anotação bem projetados, a fim de automatizar o processo de descoberta de função das sequências, da forma mais confiável possível. Em seguida, os biólogos realizam uma anotação manual, combinando os diversos resultados gerados pela anotação automática, e sua experiência e conhecimento, para atribuir funções às sequências.

A anotação e caracterização das sequências de proteínas não é uma tarefa trivial, e requer uma variedade de métodos de predição de função, além do conhecimento biológico de proteínas, para que a anotação seja confiável. Certos métodos de anotação extensivamente utilizados são baseados em: (i) homologia, investigando similaridade de sequências de nucleotídeos ou aminoácidos como *motivos* e domínios conservados; (ii) informações recuperadas de características estruturais das proteínas, como padrões espaciais de dobramento e sítios ativos de enzimas; e (iii) predição de função, investigando conservação em organismos relacionados. Em resumo, é necessário ter conhecimento do papel biológico e das características das proteínas, expresso por diferentes métodos e estratégias, para inferir uma anotação confiável.

Os problemas encontrados para anotação de proteínas são de naturezas diversas e, dentre eles, destacam-se:

- A correlação de similaridade funcional e similaridade de sequências [68] é fraca, o que pode acarretar diversos erros de transferência de anotação por similaridade de sequências. De acordo com Tramontano [81], a identidade ou similaridade de sequências pode, no máximo, garantir a existência de uma relação evolutiva entre duas proteínas, mas não garantir que possuam a mesma função;

- A transferência de anotação realizada por similaridade de sequências ou pelo conhecimento de características estruturais de domínios conservados não é suficiente para a obtenção de uma anotação confiável [68];
- Resultados provenientes de diferentes métodos e estratégias de anotação de proteínas, muitas vezes conflitantes, precisam ser combinados, o que demanda conhecimento biológico para associar corretamente uma função às sequências;
- A propagação de erro de anotação pode ser causada pelo armazenamento de anotações incertas ou imprecisas (sem evidências fortes) em bancos de dados públicos, usados extensivamente como referência de anotação em projetos ao redor do mundo.

A tarefa de anotação, por requerer um conhecimento aprofundado das características das proteínas e seu papel biológico, pode ser modelada de forma adequada em um ambiente multiagente. Uma abordagem multiagente viabiliza o uso de conhecimento especializado para realização de tarefas complexas, no qual agentes trabalham em conjunto, de maneira colaborativa, visando obter uma anotação confiável.

Neste contexto, esta dissertação propõe o BioAgents-Prot, uma ferramenta baseada em sistema multiagente (SMA) para simular o trabalho feito por um anotador humano, para auxiliar na fase de anotação de proteínas em projetos de sequenciamento.

1.1 Motivação

Deve-se notar que a tarefa de anotação de proteínas é baseada em resultados obtidos de diferentes métodos e informações de bancos de dados continuamente atualizados. Não existe uma regra clara e usual para que o biólogo possa associar adequadamente uma função a uma sequência, o que causa divergências quanto a forma de utilizar o conhecimento para realizar anotação. Portanto, representar o conhecimento que os biólogos usam para anotar em um conjunto de regras claras e bem definidas não é um procedimento fácil. Esse procedimento requer um certo tempo, de modo que um consenso entre o conhecimento e a sua representação seja obtido. Nesse processo, a arquitetura de um SMA deverá sofrer alterações até que uma proposta adequada seja obtida.

Neste contexto, em 2007, foi proposto o BioAgents [48, 49, 69, 70, 73, 79], com o objetivo de realizar anotação, tanto de proteínas quanto de RNAs não-codificadores (ncRNAs). Entretanto, como esta ferramenta foi definida para uso geral, diviculta obter anotações com maior acurácia. Em particular, como a tarefa de anotação automática de proteínas é realizada por métodos computacionais, conflitos entre os resultados podem ser produzidos, o que pode levar a predições incorretas ou anotações incompletas, dentre outros problemas. Assim, este trabalho foi motivado pela hipótese de que refinar o BioAgents com

conhecimento específico de proteínas deveria permitir obter uma anotação mais confiável, ou seja, o uso mais refinado de conhecimento de proteínas numa abordagem multiagente possibilitaria inferir funções de proteínas de forma mais confiável.

Dessa forma, a motivação deste projeto foi, dada a complexidade do processo, contribuir na etapa de anotação de proteínas de projetos transcritomas, com uma ferramenta que deveria conferir confiabilidade à anotação de sequências, com uma interface fácil de utilizar.

1.2 Problema

A arquitetura original do BioAgents não era específica para anotação de proteínas, o que dificultava usar conhecimento especializado e que atendesse, ao mesmo tempo, necessidades específicas de um projeto transcritoma para a anotação de proteínas.

1.3 Objetivos

O objetivo principal desta dissertação é definir e implementar o BioAgents-Prot, uma ferramenta multiagente, baseada em conhecimento, para anotação de proteínas em projetos transcritoma, buscando obter anotações confiáveis, a partir de regras de anotação bem definidas, mas adaptadas a um projeto específico, e investigando conservação em organismos próximos filogeneticamente.

Os objetivos específicos são:

- definir uma nova arquitetura voltada para anotação de proteínas em projetos transcritoma;
- implementar esta arquitetura utilizando o *framework* JADE [15] e o motor de inferência Drools [65];
- criar uma interface fácil de ser utilizada por um bioinformata, contemplando a possibilidade de incluir bancos de dados com anotações de organismos relacionados filogeneticamente, que poderão ser utilizados para investigar conservação de sequências;
- realizar um experimento com o fungo *Saccharomyces cerevisiae*, usando métricas para avaliar seu desempenho, comparando os resultados obtidos do BioAgents-Prot com a anotação manual das proteínas do fungo *Saccharomyces cerevisiae*.

1.4 Estrutura do documento

O presente trabalho está estruturado como segue. No Capítulo 2, apresentamos conceitos básicos relativos à anotação de proteínas, incluindo aspectos biológicos e ferramentas computacionais. Além disso, é feita uma revisão de literatura de ferramentas de anotação desenvolvidas para diferentes contextos. No Capítulo 3, descrevemos conceitos de agente inteligente e SMA, além de discutir brevemente regras de produção declarativas, usadas como modelo para representar o conhecimento e simular o raciocínio dos biólogos para anotação de proteínas. No Capítulo 4, propomos uma arquitetura baseada em SMA para anotação de proteínas e detalhamos o protótipo implementado. No Capítulo 5, discutimos um estudo de caso com o fungo *Saccharomyces cerevisiae*, e as métricas obtidas para medir o desempenho do BioAgents-Prot, quando comparado à anotação conhecida desse fungo. Finalmente, no Capítulo 6, concluímos este trabalho e sugerimos trabalhos futuros.

Capítulo 2

Anotação de proteínas

Neste capítulo abordaremos aspectos biológicos e computacionais relativos à anotação de proteínas, necessários ao entendimento deste trabalho. Na Seção 2.1 introduzimos conceitos básicos de Biologia Molecular, em particular de proteínas. Na Seção 2.2 descrevemos métodos computacionais e bancos de dados utilizados para anotar proteínas. Por fim, na Seção 2.3, apresentamos trabalhos relacionados.

2.1 Biologia Molecular

Nesta seção, descreveremos aspectos biológicos de proteínas e o Dogma Central da Biologia Molecular.

2.1.1 Proteínas

As proteínas são macromoléculas que possuem estruturas e funções biológicas diversas. Elas podem agir como enzimas catalisadoras que aceleram o processo de reações químicas, podem agir na construção de estruturas (como cabelos e unhas), podem ter funções como transporte de oxigênio, defesa do organismo ou função reguladora. Independentemente da complexidade funcional, toda proteína é formada a partir da sequência de 20 aminoácidos. O processo de formação de aminoácidos é descrito na Seção 2.1.3, enquanto o processo de construção de proteínas é descrito na 2.1.4. As estruturas das proteínas podem ser descritas em quatro níveis [21], como apresentadas na Figura 2.1:

- **Estrutura primária**: é a sequência de aminoácidos da proteína;
- **Estrutura secundária**: gerada pelo dobramento espacial de regiões próximas da estrutura primária;

- **Estrutura terciária**: gerada pela estrutura tridimensional de regiões mais afastadas das proteínas; e
- **Estrutura quaternária**: gerada pela interação espacial entre diferentes subunidades das proteínas.

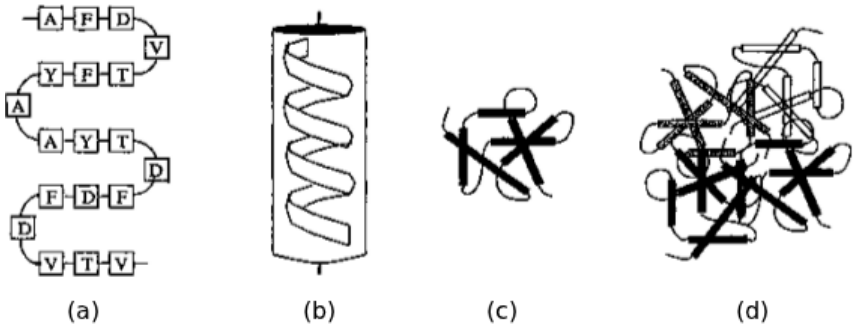


Figura 2.1: As quatro estruturas das proteínas [75]: (a) estrutura primária; (b) estrutura secundária; (c) estrutura terciária; e (d) estrutura quaternária.

As estruturas espaciais de proteínas refletem características funcionais adquiridas no decorrer da evolução, e sequências primárias similares provavelmente têm funções análogas, considerando a hipótese de que proteínas com as mesmas sequências de aminácidos foram herdadas de um ancestral comum, e possivelmente conservam a mesma função.

Os aminoácidos são gerados por ácidos nucleicos, descritos na próxima seção.

2.1.2 Ácidos nucleicos

Assim como as proteínas são formadas por sequências de aminoácidos, ácidos nucleicos são formados por nucleotídeos. Organismos vivos contêm dois tipos de ácidos nucleicos: DNA e RNA. O DNA armazena informações para codificar proteínas, sendo formado por uma sequência de nucleotídeos. Cada nucleotídeo é composto por um grupo fosfato, uma pentose (desoxirribose) e uma base nitrogenada. O DNA contém quatro tipos de bases nitrogenadas, sendo elas: adenina (A), timina (T), citosina (C) e guanina (G).

A molécula de DNA consiste de duas cadeias (fitas) que enovelam ao redor do mesmo eixo, formando uma dupla hélice [85]. Essa formação espacial de dupla hélice ocorre por meio de ligações entre bases nitrogenadas complementares A-T e C-G.

De forma diferente, no RNA, a timina (T) é substituída por uracila (U) e sua pentose é a ribose. Além disso, a molécula de RNA é formada em geral por uma única fita, podendo assumir diversas funções e formas no organismo. A Figura 2.2 ilustra a estrutura das moléculas de DNA e RNA.

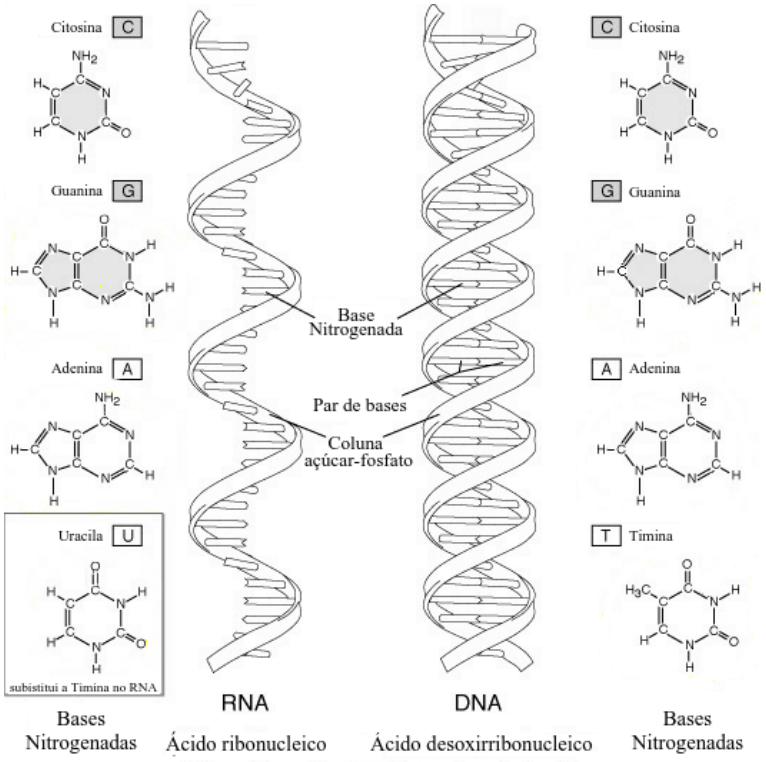


Figura 2.2: Estruturas das moléculas de DNA e RNA (adaptado de [47]).

2.1.3 Gene e o código genético

Um cromossomo é uma longa sequência de DNA e, ao longo dessa sequência, encontramos regiões chamadas genes. Os genes, formados por sequências de nucleotídeos, carregam o código genético necessário para a produção de proteínas. Cada aminoácido é formado por uma sequência de códons (sequências de três nucleotídeos). São conhecidos 64 códons que representam o código genético (Figura 2.3). Um dos 64 códons representa o início de tradução (AUG) - o aminoácido metionina (Met), e três códons representam condições de parada de tradução.

2.1.4 O Dogma Central da Biologia Molecular

O Dogma Central da Biologia Molecular, como originalmente definido por Watson & Crick [85], modela o processo da síntese de proteínas. Como dito, um gene contém a informação para a síntese de proteína, sendo essa informação carregada por um RNA intermediário, denominado de RNA mensageiro (mRNA). O mRNA, com auxílio do RNA ribossomal (rRNA) e do RNA transportador (tRNA), é traduzido em proteína (Figura 2.4).

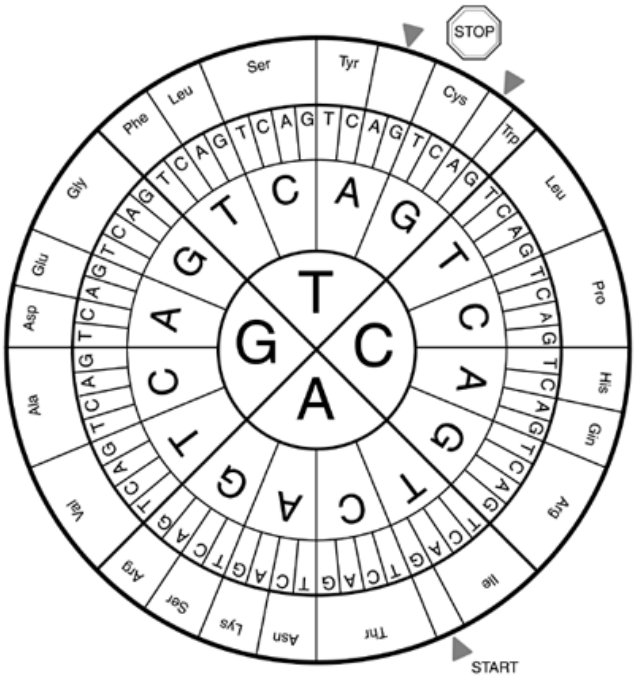


Figura 2.3: Representação da tabela do código genético [16].

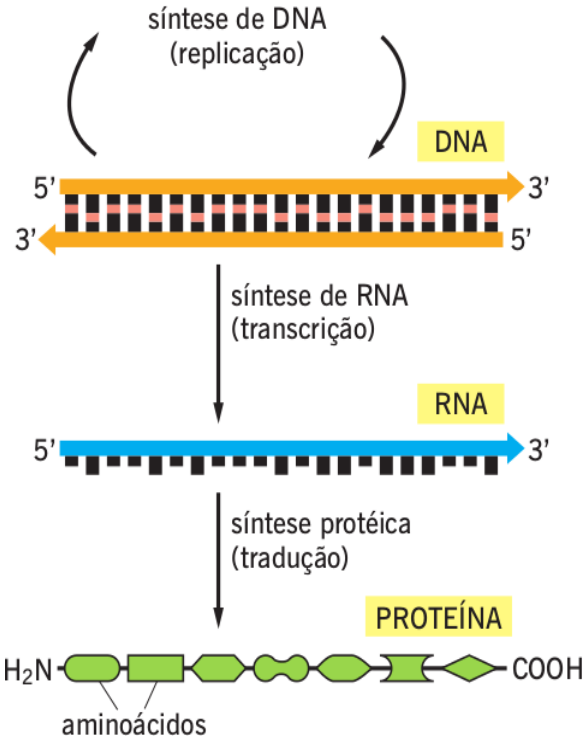


Figura 2.4: Dogma Central da Biologia Molecular [10]. Os processos de replicação, transcrição e tradução são conhecidos como expressão gênica.

Em eucariotos, a transcrição do DNA produz um pre-mRNA composto de *íntrons* e *éxons*. O processo de amadurecimento do pre-mRNA envolve a remoção dos íntrons ao longo de sua sequência, transformando-se em um mRNA maduro, composto de apenas *éxons*. O processo de remoção de introns é denominado de excisão (*splicing*), sendo esse processo realizado por pequenos RNAs nucleares (*small nuclear ribonucleic acids - snRNAs*). Em seguida, o mRNA maduro é transportado até o citoplasma para tradução em proteínas. De forma diferente, organismos procariotos não contêm íntrons em seu DNA. Logo, não ocorre o processo de *splicing*.

Os processos acima descrevem como a informação armazenada no DNA é expressa [86]. Isso envolve a transcrição de parte da sequência de DNA na forma de uma molécula de RNA, que é então usada como molde para tradução em proteína. Mas nem todos os genes são expressos em todas as células continuamente. Pelo contrário, a vida de um organismo depende da habilidade das células de expressar seus genes em diferentes combinações em tempos diferentes e em locais diferentes. Por exemplo, uma bactéria expressa apenas alguns de seus genes em um determinado momento, assegurando que pode tanto construir as enzimas necessárias para metabolizar os nutrientes disponíveis no meio, quanto construir outras enzimas quando esses nutrientes não estão disponíveis. O desenvolvimento de organismos multicelulares é outro exemplo interessante de *expressão diferencial de genes*. Geralmente, todas as células humanas contêm os mesmos genes, mas diferentes conjuntos de genes são expressos para formar diferentes células. Então, uma célula muscular expressa um conjunto de genes diferentes (pelo menos parcialmente) daqueles expressos por um neurônio, uma célula epitelial, e assim por diante. Essas diferenças ocorrem mais comumente em nível de transcrição, em particular, a iniciação da transcrição sofre regulação.

Em bactéria, temos alguns casos simples de regulação transcricional. O *lac operon*, um grupo de genes que codificam proteínas necessárias ao metabolismo dos genes de lactose e açúcar, é transcrito apenas quando o açúcar está disponível no meio de crescimento. Neste caso, os genes podem ser ativados e reprimidos em resposta aos sinais diferentes.

Em eucariotos, mecanismos de ativação e repressão transcricional são semelhantes aos de bactéria, sendo alguns mecanismos conservados e outros apresentando novas características, como efeitos de posicionamento, remodelagem e modificação de nucleossomo.

De forma geral, os exemplos acima mostram exemplos de proteínas que exercem regulação direcionada por ativação e repressão. Existem também RNAs reguladores, que podem ativar e reprimir expressão de genes em bactérias e eucariotos. Isso inclui mecanismos conhecidos há bastante tempo, como atenuação do *operon triptofano*, e outros descobertos recentemente, como RNA interferente (RNAi) e microRNAs em eucariotos mais complexos.

Outros exemplos de regulação são: genes regulados de forma a permitir especificidade a uma célula (diferenciação) e formação de padrão (morfogênese) em um grupo de células idênticas geneticamente, por exemplo, células encontradas no desenvolvimento embrionário; a diferença morfológica ou comportamental entre organismos próximos filogeneticamente é devido não à mudança de genes, mas às diferenças em quando e onde os genes são expressos dentro de cada organismo durante o desenvolvimento.

Recentemente, os genomas sequenciados mostraram que a maioria dos animais (por exemplo) têm essencialmente os mesmos genes - camundongo, humano ou moscas. Esses exemplos mostram o importante papel dos genes de regulação (a maioria deles regulação transcricional) nos produtos de um genoma.

2.2 Métodos, ferramentas e bancos de dados

A partir do surgimento dos sequenciadores de alto desempenho [39], o número de sequências não caracterizadas vêm aumentando em ritmo acelerado. Como dito anteriormente, em projetos genoma e transcritoma, a descoberta de funções de sequências é feita numa fase chamada de anotação. A anotação automática é feita pela execução de programas (por exemplo, que buscam similaridades de sequências, dentre outros) e consulta à bancos de dados (por exemplo, contendo sequências e funções já determinadas). A partir dessas informações, são realizadas anotações por biólogos, sendo esta etapa denominada de anotação manual, que envolve muito tempo e diferentes análises, e hoje não são capazes de acompanhar a produção em larga escala das novas sequências. Como consequência, novos métodos e abordagens de predição de função vêm surgindo como solução para automatização do processo de anotação e caracterização destas novas sequências.

Esta seção apresenta diversas abordagens de predição de função, e exemplos de ferramentas e bancos de dados de proteínas, estando o texto dividido de acordo com a classificação de métodos proposta por Pandey e co-autores [61].

De acordo com Bork e co-autores [20], "função" é uma definição que faz sentido em um determinado contexto. Shrager [76] argumenta que a função pode ser descrita em diferentes níveis, variando desde funções bioquímicas, processos biológicos e vias metabólicas, até o nível de órgãos e sistemas do organismo. Consequentemente, as proteínas são descritas em diferentes níveis de especificidade funcional, que descrevem seu papel biológico no organismo. Bork e co-autores [20] categorizaram os tipos de funções que uma proteína pode realizar em três níveis (Figura 2.5), descritos a seguir:

1. **Função molecular**: As funções bioquímicas realizadas por uma proteína, tal como formação de ligantes, catálise de reações bioquímicas e mudanças conformacionais;

2. **Função celular**: Muitas proteínas juntam-se para realizar funções fisiológicas complexas, para manter um funcionamento adequado de vários componentes do organismo, tais como o funcionamento das vias metabólicas e transdução de sinal; e

3. **Função fenotípica**: A integração de vários subsistemas fisiológicos e a sua interação com vários estímulos do ambiente determinam as propriedades fenotípicas e o comportamento do organismo.

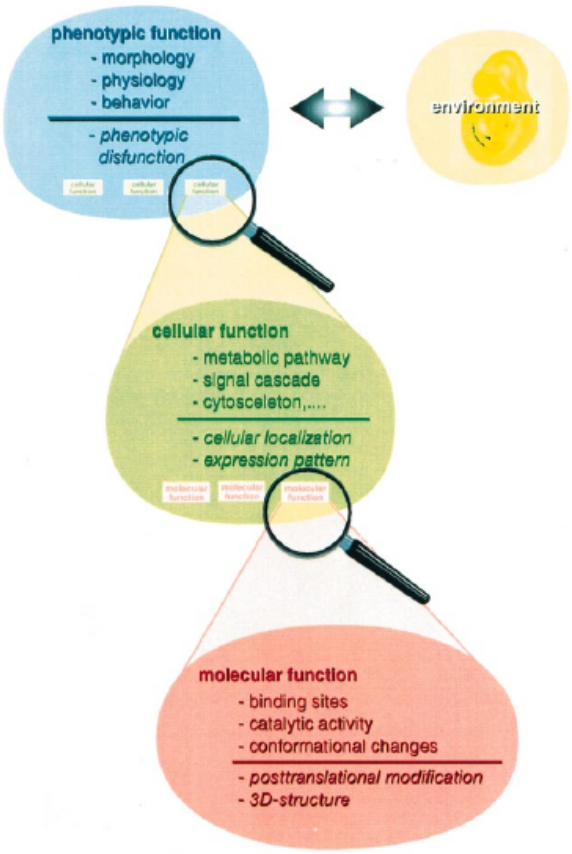


Figura 2.5: Caracterização conceitual da função, definida por Bork e co-autores [20].

Entretanto, esta não é a única categorização proposta. Podemos citar a classificação do *Gene Ontology* (GO), que categoriza as funções das proteínas em componente celular, função molecular e processo biológico [13].

2.2.1 Transferência de anotação baseada em homologia

Consultas por similaridades de sequências permitem inferir proteínas ou genes "homólogos", isto é, sequências similares (que possuem aproximadamente a mesma sequência de aminoácidos), estatisticamente significativas, que possivelmente foram herdadas de um

ancestral comum, indicam homologia. Este processo tem como hipótese de que os organismos herdam características de seus ancestrais no processo de evolução, preservando informações genéticas de sua linhagem, em particular funções biológicas importantes [24, 52]. O método de buscar similaridades é extensivamente utilizado para predição de função de proteínas não caracterizadas, conhecido como transferência de anotação baseada em homologia.

Segundo Pearson [62], a busca por similaridades de sequências é uma estratégia efetiva para inferir homólogos, porém a inferência da similaridade funcional a partir da homologia é mais difícil. A limitação mais significativa deste método decorre da evolução divergênte em resposta à pressão seletiva [34, 81], onde uma duplicação de um determinado gene pode desenvolver uma nova função, não garantindo que a predição de função por homologia esteja correta. Uma outra limitação está relacionada à identificação de sequências homólogas formadas por múltiplos domínios [81]. Diferentes configurações de domínios nessas sequências produzem proteínas distintas com funções diferentees e, portanto, torna-se difícil inferir por homologia que as sequências compartilham a mesma função. Como consequência, a transferência de anotação baseada em homologia não é totalmente confiável e, muitas vezes, pode ser inadequada, o que pode causar propagação de erro nos bancos de dados [58, 61, 81]. A Figura 2.6 mostra um exemplo.

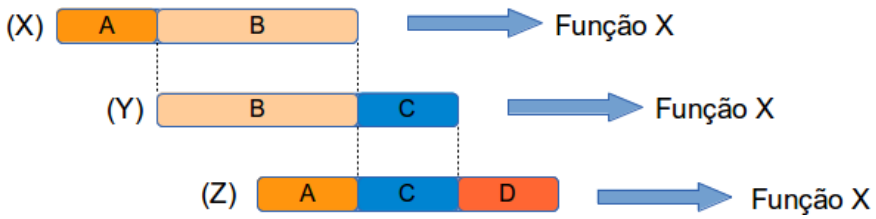


Figura 2.6: Propagação de erro por transferência de anotação baseada em homologia. Esta figura ilustra uma proteína com uma função Y, que possui domínios B e C, sendo anotada por homologia com a função da proteína X, que possui domínios A e B. Em seguida, uma proteína de função Z, que possui os domínios A, C e D, é anotada por homologia com a mesma função da proteína X, propagando o erro de anotação para as demais proteínas. Figura adaptada de [81].

De acordo com Pandey e co-autores [61], a similaridade de sequências tem sido intensamente utilizada para predição de função com as seguintes abordagens:

- **Abordagem baseada em homologia**: Esta abordagem nem sempre é precisa, simplesmente por causa da fraca correlação entre a similaridade de sequências e a similaridade funcional [61, 68]. Por este motivo, diversas estratégias buscam combinar resultados de diversas fontes para realizar uma predição por função mais acu-

rada, tais como investigar conjuntos (*clusters*) de sequências com funções similares e pesquisar categorias funcionais do *Gene Ontology* (GO), dentre outras fontes;

- **Abordagem baseada em subsequências**: Nesta abordagem, utilizam-se estratégias para identificar regiões ou subsequências relevantes, tais como *motivos* e domínios conservados, capazes de indicar uma determinada função. Essas regiões geralmente são extraídas por consenso obtido do alinhamento múltiplo de diversas sequências com as mesmas funcionalidades, mas também podem ser obtidas a partir de uma consulta iterativa na busca por homólogos distantes;
- **Abordagem baseada em características**: Esta abordagem utiliza algoritmos de classificação para criar modelos de classes funcionais das proteínas, obtidas a partir de suas características comuns. Os classificadores geralmente utilizados são SVM, redes neurais e inferência Bayesiana. Estes modelos são utilizados para predição de função.

Algumas ferramentas utilizadas para buscas por similaridade empregadas nessas abordagens são:

- *Basic Local Alignment Search Tool* (BLAST) [11, 56] busca por alinhamentos locais entre sequências de nucleotídeos e proteínas, comparando e calculando a significância estatística de cada sequência em um banco de dados. O Blast tem heurísticas que aceleram a busca por similaridades locais, entretanto não garante um alinhamento ótimo. O cálculo de similaridade é auxiliado por uma matriz de substituição, que modela probabilidades de ocorrência de mutações. Por exemplo, uma matriz padrão bastante utilizada é a BLOSUM62;
- *BLAST Like Alignment Tool* (BLAT) [18, 40] é uma ferramenta que busca alinhamento local, mas com um método diferente. O BLAT tenta localizar as sequências de entrada em um banco de dados, assumindo que as sequências de entrada tenham alta similaridade com as sequências do banco de dados. Assim, apenas alinhamentos com alta similaridade serão considerados. Enquanto o foco do BLAST é encontrar alinhamentos locais gerais, o que permite identificar alinhamentos remotos e similares, indexando as sequências de entrada para posterior consulta com os bancos de dados, o BLAT indexa o genoma inteiro para identificar os possíveis alinhamentos locais com a sequência de entrada. Indexar o banco de genoma, em vez das sequências de entrada, produz alinhamentos mais rapidamente, na maioria dos casos.
- *BlastProDom* [74] é um *script* escrito na linguagem Perl, utilizado para consultar e mapear as famílias de domínios do banco ProDom nas sequências de entrada. O *script*, atualmente obsoleto, apresenta erros de execução e utiliza a ferramenta

BLASTALL ou *Legacy BLAST*, primeira versão do BLAST. O objetivo do *script* é realizar uma consulta por similaridade com os domínios do banco de dados ProDom e, em seguida, realizar um filtro identificando os melhores alinhamentos de famílias diferentes e removendo os alinhamentos que pertencem à mesma família. O resultado final é obter um mapeamento de início e final de diferentes famílias de domínios às sequências de entrada;

- *Hmmer* [29, 30] utiliza perfis de modelos ocultos de Markov (*profile Hidden Markov Models* – *profile-HMM*) para realizar a comparação de sequências, sendo utilizado para consulta por homólogos em bancos de dados de sequências de proteínas ou DNA. Hmmer visa ser significativamente preciso, além de ser capaz de detectar homólogos remotos, devido aos seus modelos. Entretanto, nas versões anteriores, seu custo computacional era $100x$ mais lento que a consulta do BLAST em bancos de proteínas, e $1.000x$ mais lento que a consulta do BLAST em bancos de DNA. A partir da versão $3.x$, Hmmer passou a ser tão rápido quanto o BLAST para consultas em bancos de proteínas.

Bancos de dados de proteínas geralmente são utilizados por ferramentas de similaridade de sequências para consulta de sequências "homólogas", permitindo inferir por homologia uma determinada anotação. Os principais bancos de proteínas são descritos a seguir:

- *Swiss-Prot* [19] é um banco de sequências de proteínas que apresentam anotações curadas e de alta qualidade, contendo nomeclaturas padronizadas, *links* para bancos de dados especializados e mínima redundância;
- *TrEMBL* [19] é um banco suplementar ao banco Swiss-Prot, formado por sequências de proteínas que possuem anotações automáticas;
- *RefSeq* [66] é um banco não redundante de sequências de genes, transcritos e proteínas, provido e curado pelo NCBI. As sequências anotadas incluem regiões de codificação, domínios conservados, variações, referências cruzadas entre bancos, dentre outras características;
- *NR* [2] é um banco de sequências de proteínas não redundante, construído a partir das entradas dos bancos GenPept, Swiss-Prot, PIR, PDF, PDB e RefSeq.
- *ProDom* [74] é um banco que reúne um conjunto de famílias de domínios das proteínas, que foram geradas automaticamente a partir das sequências dos bancos Swiss-Prot e TrEMBL;

- *Pfam* [67] é um banco de famílias de proteínas construídas com HMM, onde as famílias são constituídas por um conjunto de proteínas que compartilham um nível significativo de similaridade, sugerindo assim homologia. Pfam contém dois tipos de famílias: famílias de alta qualidade curadas manualmente (Pfam-A) e famílias geradas automaticamente (Pfam-B). As famílias Pfam devem representar unidades funcionais (domínios) que, quando combinados de diferentes maneiras, podem gerar proteínas com funções únicas. As famílias podem conter domínios não caracterizados, identificados no Pfam como *"Domain of Unknown Function"* (DUF). As famílias não caracterizadas são identificadas no Pfam como *"Uncharacterized Protein Family"* (UPF).

2.2.2 Anotação baseada em estrutura das proteínas

A estrutura das proteínas determina boa parte de suas características funcionais, tais como a localização celular, os tipos de ligantes e as interações com outras proteínas. Um exemplo, comumente apresentado nas estruturas das proteínas, são os sítios ativos das enzimas. Esses sítios constituem partes de uma enzima nas quais o substrato da reação liga-se a si mesmo, participando diretamente nos mecanismos de reação catalisadora, refletindo a sua função. Pode-se observar neste exemplo que a estrutura da proteína reflete sua função biológica, podendo ser de utilidade na inferência de função.

Diversas abordagens para predição de função a partir da estrutura das proteínas foram divididas e agrupadas por Pandey e co-autores [61] em quatro grandes grupos:

- **Abordagens baseadas em similaridade**: Dada a estrutura da proteína, estas abordagens identificam a proteína com a estrutura mais similar, utilizando técnicas de alinhamento estrutural e transferindo sua anotação para uma proteína não caracterizada;

- **Abordagens baseadas em *motivos***: Estas abordagens buscam identificar *motivos* tridimensionais, que são subestruturas conservadas em um conjunto de proteínas funcionalmente relacionadas. Os métodos estimam um mapeamento entre a função de uma proteína e os *motivos* estruturais que ela contém. Esse mapeamento é utilizado para predizer funções das proteínas não caracterizadas;

- **Abordagens baseadas em superfície**: As interações entre moléculas levam a funções bioquímica e ocorrem no nível dos aminoácidos. No entanto, essas interações ocorrem em muitos casos, devido à complementariedade das superfícies moleculares das proteínas. As características das superfícies podem indicar muitas funções das proteínas, e as informações fornecidas por essas estruturas têm sido utilizadas em vários métodos computacionais para predição de função das proteínas;

- **Abordagens baseadas em aprendizado**: Estas abordagens empregam métodos de classificação, tais como SVM e *k-nearest neighbor (K-NN)*, para identificar a classe funcional mais apropriada para uma proteína, por meio das características estruturais mais relevantes.

Os bancos de dados de estruturas das proteínas não são tão diversificados quanto os bancos de sequências. Dentre eles, os mais conhecidos, geralmente utilizados para predição de função a partir de estruturas, são:

- *Protein Data BANK* (PDB) [17] é o banco mais popular de estruturas 3D de proteínas experimentalmente determinadas. Ferramentas de análise estrutural com o banco PDB estão disponíveis no *site* do PDB, por meio de uma interface web;
- *Structural Classification of Proteins* (SCOP) [12] é um banco de proteínas conhecidas, ordenadas de acordo com seu relacionamento estrutural e evolutivo. Os domínios das proteínas neste banco são classificados hierarquicamente em famílias, superfamílias, dobramentos e classes;
- *Class, Architecture, Topology and Homologous superfamily* (CATH) [59] provê uma classificação hierárquica de domínios de proteína baseada em seus padrões de dobramentos. Domínios são obtidos de estruturas de proteínas depositadas no PDB.

2.2.3 Anotação baseada em sequências genômicas

Várias abordagens têm sido propostas para realizar o objetivo de obter dados a partir de associações funcionais do genoma e, posteriormente, possível predição de função. Estas abordagem enquadram-se em uma das três categorias a seguir:

- **Transferência de anotação baseada em homologia em todo o genoma**: Esta categoria consiste simplesmente em consultar proteínas homólogas em bancos de dados e, em seguida, transferir a anotação funcional da sequência que obtiver melhor similaridade;
- **Abordagens baseadas em vizinhança gênica**: Estas abordagens são baseadas na hipótese de que as proteínas, cujos genes correspondentes são localizados "próximos" uns dos outros em um genoma, estão relacionadas funcionalmente. Portanto, torna-se uma estratégia viável para inferir associações funcionais entre genes e suas proteínas correspondentes;
- **Abordagens baseadas em fusão gênica**: Estas abordagens buscam descobrir pares ou conjuntos de genes que são "unidos" para formar um único gene em outro genoma, partindo da hipótese (suportada por evidências estruturais e bioquímicas) de que estes conjuntos de genes estão relacionados funcionalmente.

2.2.4 Anotação baseada em dados filogenéticos

A evolução de uma espécie de organismos para outra tem sido uma área de pesquisa ativa na Biologia, desde Darwin [24] e, posteriormente veio a constituir uma área de pesquisa denominada de filogenia. Desde então, diversos estudos foram conduzidos em filogenia, dentre eles, o mais relevante para esta dissertação são estudos que buscam descobrir funções dos genes e proteínas e suas ligações funcionais, por meio de perfis ou árvores filogenéticas. Nesta seção, baseada em Pandey e co-autores [61], as abordagens são divididas em três categorias:

- **Abordagens utilizando perfis filogenéticos**: Estas abordagens partem da hipótese que proteínas com perfis filogenéticos comuns estão funcionalmente relacionadas. Os métodos utilizam formas de mensurar a similaridade entre perfis filogenéticos;
- **Abordagens utilizando árvores filogenéticas**: Árvores filogenéticas incorporam um conhecimento de evolução genética mais rico do que perfis. Como consequência, diversos métodos exploram o conteúdo de árvores filogenéticas para predição de função. Muitas delas utilizam mineração de dados e aprendizado de máquina para realizar esta tarefa;
- **Abordagens híbridas**: Abordagens recentes utilizam técnicas baseadas em SVM, que combinam informações evolutivas de perfis e árvores filogenéticas.

Uma ferramenta utilizada para construir árvores filogenéticas é o *Clustal* Ω [25, 77], que utiliza HMM como base para realizar alinhamentos múltiplos de sequências. Sua precisão em um pequeno número de sequências é similar a outros algoritmos de alta qualidade e, para um conjunto muito grande de sequências, Clustal Ω supera os demais algoritmos em tempo e qualidade. Clustal Ω é capaz de realizar um alinhamento múltiplo de 190.000 sequências em apenas algumas horas [77].

2.3 Trabalhos relacionados

Diferentes técnicas computacionais são empregadas em diferentes tarefas de bioinformática, incluindo a análise e predição de funções das proteínas. Em particular, são utilizadas técnicas de Inteligência Artificial, tais como SMA, mineração de dados e aprendizado de máquina.

Projetos transcritoma e reguloma, bem como projetos metagenômicos, sequenciados com técnicas de alto desempenho, têm sido anotados com sucesso por meio de méto-

dos tradicionais, ligeiramente modificados para lidar adequadamente com os resultados produzidos por esses sequenciadores [38, 55, 78, 84].

Multi-Agent System to Support Functional Annotation - MASSA [90] integra o conhecimento biológico em um ambiente multiagente para apoiar a anotação funcional das proteínas. Os principais componentes do núcleo do sistema são *MASSAPipe* e *MASSAInference*, ambos coordenados por um agente controlador. *MASSAPipe* gerencia o *pipeline* de anotação, determinando quais agentes de ferramentas deverão ser executados para que possam extrair dos bancos de dados informações relevantes de anotação. Por outro lado, *MASSAInference* gerencia os agentes de inferência, onde cada um provê como mecanismo de raciocínio um motor de inferência e uma base de regras, utilizados para inferir uma anotação, a partir de um conjunto de informações coletadas pelo *MASSAPipe*. Esses agentes também consultam termos do *Gene Ontology* para refinar a anotação sugerida.

Feature Architecture Comparison Tool - FACT [41] é uma ferramenta que combina diferentes características das proteínas para predição de anotação funcional, por exemplo, domínios funcionais, elementos da estrutura secundária e propriedades composicionais. FACT apresenta resultados que podem identificar equivalentes funcionais, mesmo quando as sequências compartilham baixa similaridade.

Forslund e Sonnhammer [33] desenvolveram dois modelos que realizam predição de anotações dos termos GO a partir dos domínios das proteínas: um modelo baseado em regras e um modelo probabilístico. O primeiro generaliza e estende o mapeamento do *Pfam2GO* a vários domínios. O segundo modelo utiliza uma representação probabilística entre combinações de domínios (que podem codificar diferentes funções) e anotações dos termos GO. Os resultados apresentaram boas melhorias em relação ao Pfam2GO e melhor precisão em relação à anotação recuperada do melhor hit do BLAST. Além disso, o modelo probabilístico apresenta, em alguns casos, uma melhor correlação em relação à anotação obtida do melhor *hit* e ao Pfam2GO.

Orro e co-autores [60] propuseram uma abordagem multiagente para análise de função das proteínas em uma infraestrutura de grade foi desenvolvida como proposta de apoio para classificação de proteínas. Nesta abordagem, o ambiente multiagente é distribuído em uma grade (*grid*) para classificação de proteínas, baseado-se em um *pipeline* filogenômico.

Electronic Annotation-EAnnot [27] é uma ferramenta desenvolvida originalmente para o projeto genoma humano. O software combina ferramentas para extrair e analisar grandes volumes de dados com o intuito de realizar anotação automática e inferência de genes. Entre outros, *EAnnot* utiliza informações contidas em *mRNAs*, *ESTs* e alinhamentos das proteínas para identificar pseudogenes.

Environment for Automatic Annotation and Comparison of Genomes - A3C [72] é baseado em uma arquitetura multiagente, dividida em dois níveis. O Nível 1 tem o objetivo

de integrar tarefas relacionadas à fase de anotação, utilizando ferramentas para anotação automática das proteínas. O Nível 2 utiliza algoritmos de comparação genômica para extração de informações relevantes do Nível 1. O objetivo do *A3C* é identificar relações entre diferentes organismos. Este procedimento é realizado pela obtenção de características particulares dos ornganismos estudados, utilizando informações de organismos já conhecidos.

Agent-based environmenT for aUtomatiC annotation of Genomes - ATUCG [57] possui uma arquitetura de agentes com uma interface interativa com o usuário, capaz de auxiliar o biólogo no processo de re-anotação. Neste processo, a informação das sequências já anotadas são revisadas e comparadas a novos modelos de dados, na busca de obter características e informações sobre as sequências e, se necessário, estas sequências serão re-anotadas.

Finalmente, *BioMAS* [26] utiliza SMA para fase de anotação automática do vírus da herpes. Seu objetivo é a extração de informações contidas nos bancos de dados públicos e, em seguida, realização da anotação automática.

A Tabela 2.1 apresenta um resumo comparativos dos trabalhos relacionados citados acima.

Tabela 2.1: Resumo dos trabalhos relacionados.

Ferramenta	**Abordagem de anotação**	**Método**
MASSA	*sequência* $\rightarrow$ *homologia* $\rightarrow$ *função*	SMA baseado em conhecimento
FACT	*sequência* $\rightarrow$ *estrutura* $\rightarrow$ *função*	Não descrito
Forslund e Sonnhammer	*sequência* $\rightarrow$ *domínios* $\rightarrow$ *função*	modelo probabilístico e modelo baseado em regras
Orro e co-autores	pipeline filogenômico	SMA distribuído em grade
EAnnot	*genes* $\rightarrow$ *homologia* $\rightarrow$ *função*	Não descrito
A3C	*genes* $\rightarrow$ *homologia* $\rightarrow$ *função*	SMA
ATUCG	re-anotação	SMA baseado em ambiente
BioMAS	*genes* $\rightarrow$ *homologia* $\rightarrow$ *função*	SMA

Capítulo 3

Sistema multiagente

Este capítulo apresenta noções básicas de SMA, necessárias para o entendimento deste trabalho. Dentre essas noções, é introduzido na Seção 3.1 o conceito de agente inteligente e SMA. Na Seção 3.2, é apresentado o conceito de agente baseado em conhecimento, incluindo definições da arquitetura interna dos agentes, representação do conhecimento, seus mecanismos de inferência e uma breve descrição dos motores de inferência baseado em regras. Na Seção 3.3, são descritas algumas especificações recomendadas para SMA, categorizadas em cinco eixos pela FIPA[1]. Na Seção 3.4, é descrito brevemente algumas ferramentas de desenvolvimento de SMA. Finalmente, na Seção 3.5, é realizado uma discussão sobre os temas abordados neste capítulo.

3.1 Agente inteligente e SMA

Agentes são entidades computacionais capazes de realizar ações autônomas em um determinado ambiente, na tentiva de alcançar seus objetivos [71, 83, 88]. Um agente que busca sempre otimizar sua medida de performance é denominado "agente racional" [83]. De acordo com Weiss [87], "agentes inteligentes" são aqueles que perseguem seus objetivos e executam tarefas de tal forma que sua medida de performance esteja sendo otimizada, ou seja, são agentes flexíveis que agem racionalmente às circunstâncias de seu ambiente, limitados pela informação obtida e pelas capacidades de perceção e ação. De maneira geral, Russell & Norvig [71] definem um agente como uma entidade capaz de perceber seu ambiente a paritir de sensores e de agir sobre esse ambiente por intermédio de atuadores, como ilustrado na Figura 3.1.

Segundo Wooldridge & Jennings [89], um comportamento autônomo e flexível permite ao agente exibir controle sobre suas ações e seu estado interno, sendo esta flexibilidade caracterizada como segue:

[1] *Foundation for Intelliget Physical Agents* (FIPA).

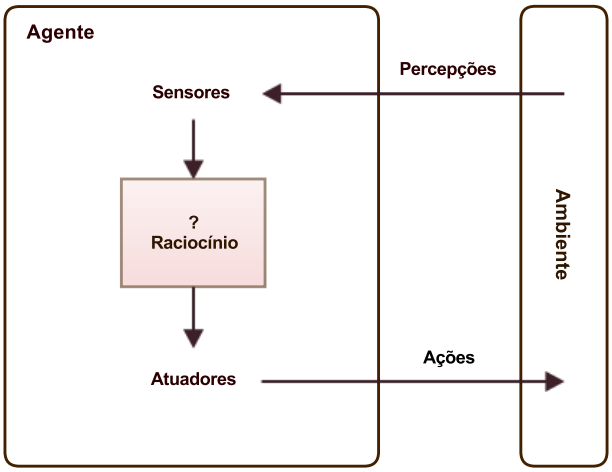


Figura 3.1: Arquitetura geral de um agente [71].

- **reatividade**: agentes percebem o ambiente e respondem às mudanças que ocorrem no mesmo;
- **proatividade**: agentes são capazes de exibir comportamento dirigido à objetivos tomando iniciativas e reconhecendo oportunidades; e
- **interatividade**: em respeito às habilidades sociais, agentes são capazes de interagir uns com os outros (e até mesmo com seres humanos) através de algum mecanismo de lingaugem de comunicação.

Segundo Wooldridge [88], um sistema multiagente consiste de um número de agentes que interagem uns com os outros, geralmente por troca de mensagens. Agentes podem ser afetados tanto por outros agentes quanto por intervenção humana, na busca de alcançar seus objetivos e realizar tarefas. Em um ambiente compartilhado, mecanismos de coordenação são propostos para evitar o estado das coisas consideradas desejáveis ou indesejáveis por um ou mais agentes, na tentativa de coordenar objetivos e tarefas dos agentes.

Duas formas contrastantes de coordenação são [87]:

i) **cooperação**: agentes trabalham em conjunto de forma a maximizar as possibilidades de alcançar objetivos comuns; e

ii) **competição**: agentes trabalham sozinhos agindo uns contra os outros, pois seus objetivos são individuais.

Agentes cooperativos são classificados como agentes de interesses comuns, os quais se agrupam para alcançar objetivos que não podem ser realizados individualmente, de tal

forma que o sucesso no alcance dos objetivos decorrerá das ações do grupo como um todo. Por outro lado, agentes competitivos são classificados como agentes de interesses próprios, os quais buscam maximizar seu próprio benefício às custas dos outros, tal que o sucesso de um implicará no fracasso dos outros.

3.2 Agente baseado em conhecimento

Agentes lógicos são projetados para formar representações do mundo, usar mecanismos de inferência para derivar novas representações sobre o mundo e utilizar essas novas representações para deduzir o que fazer [71]. Essas representações do mundo compõem a base de conhecimento (BC) do agente, a qual é formulada por um conjunto de sentenças lógicas e fatos. Os agentes baseados em conhecimento podem se beneficiar do conhecimento expresso (geralmente por regras declarativas), combinando e recombinando informações para atender uma infinidade de propósitos. Esses agentes devem ser capazes de registrar novas informações na BC (TELL), consultar o que se conhece (ASK) e, remover informações da BC (RETRACT). A Figura 3.2 ilustra uma arquitetura de um agente baseado em conhecimento com estas capacidades e características.

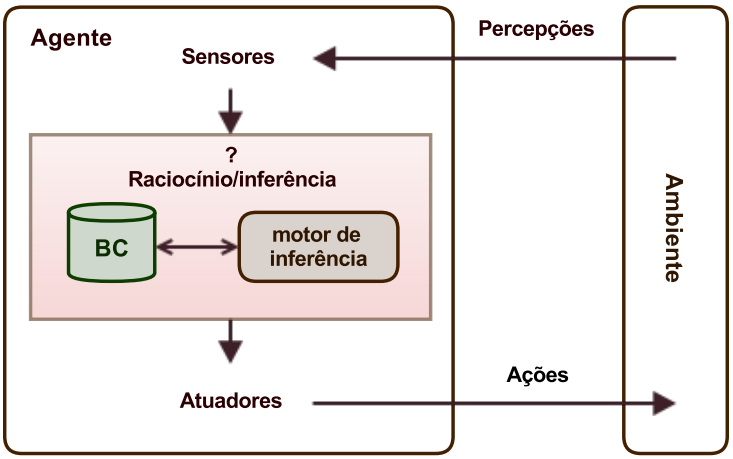


Figura 3.2: Arquitetura de um agente baseado em conhecimento: definida a partir da arquitetura abstrata de Russell & Norvig (Figura 3.1) e de um modelo genérico de motor de inferência. Motores de inferência são mecanismos de inferência utilizados em conjunto com uma base de conhecimento para realizar asserções, derivar novas representações do mundo e deduzir possíveis coclusões ou ações que um agente pode assumir.

A arquitetura apresentada na Figura 3.2 inclui um motor de inferência genérico com suas funcionalidades básicas. Existem diversos modelos de representação de conhecimento que exige um mecanismo de raciocínio específico associado, tais como regras declarativas, ontologias, árvores de decisão e entre outros. Poucos motores de inferência possuem

mecanismos de inferência híbrido que combinam inferência de mais de um modelo, como por exemplo o Jena [1], descrito em seguida na Seção 3.2.3, que possui mecanismos de inferência tanto para regras quanto para ontologias.

Diferentemente de outras arquiteturas, agentes baseados em conhecimento não possuem um mecanismo arbitrário para calcular ações. Devido às definições TELL, ASK e RETRACT, agentes deste tipo adaptam-se a uma descrição no *nível de conhecimento*, em que se deve especificar apenas o que o agente conhece e quais são suas metas para que possam, em seguida, deduzir o que fazer. Além destas capacidades, podemos fornecer aos agentes baseado em conhecimento mecanismos que lhes permitam aprender por si mesmos, o que lhes confere a capacidade de identificar características ocultas do ambiente que são desconhecidas em seu conjunto de regras e, em seguida, criar e associar novas sentenças e relações (regras, predicados e conceitos ontológicos) descobertas à BC, a partir de uma série de percepções. Desse modo, o agente pode ser completamente autônomo.

Russel & Norvig [71] definem um agente baseado em conhecimento genérico com os procedimentos mostrados no Código 3.1 (código com adaptações).

Código 3.1: Procedimentos de um agente baseado em conhecimento genérico.

```
1 PERCEIVE (entrada) : percebe uma entrada
2 TELL (BC, entrada) : informa à BC o que é percebido (entrada)
3 ação ← ASK (BC, inferência) : consulta à BC para deduzir/inferir uma ação
4 TELL (BC, ação) : informa à BC a ação tomada
5 RETORNA ação : realiza ação
```

3.2.1 Representação do conhecimento

Para criar um agente baseado em conhecimento, é necessário inicialmente elaborar sentenças e regras que representem o conhecimento que o projetista ou especialista tem do ambiente. O processo de representação do conhecimento do especialista em um conjunto de regras bem definido decorre de várias entrevistas e é conhecido como engenharia de conhecimento (Figura 3.3).

Nesse processo, um engenheiro do conhecimento realiza entrevistas com um especialista, a fim de extrair e representar seu conhecimento em um conjunto de sentenças e regras claras e bem definidas. Para obter um consenso entre o conhecimento do especialista e sua representação em regras, é necessário que o engenheiro valide suas regras nas entrevistas subsequentes, até que uma BC concisa e confiável seja obtida.

A partir de uma base de conhecimento bem definida, um agente baseado em conhecimento poderá ser inserido em um ambiente para o qual ele foi projetado para operar. Em outras palavras, o agente estará pronto para receber percepções de seu ambiente,

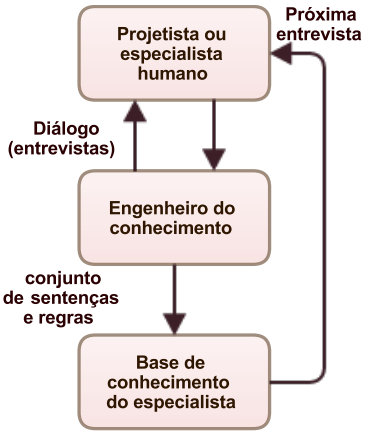


Figura 3.3: Processo de elaboração da base de conhecimento do projetista ou especialista (adaptado de Giarratano [35]).

informando (com TELL) o que ele precisa conhecer e questionando (com ASK) que ações serão deduzidas, assim como os procedimentos descritos no Código 3.1.

3.2.2 Raciocínio lógico baseado em regras

A Lógica Computacional, como utilizada na área de Inteligência Artificial, é uma *linguagem de representação de conhecimento* de um agente, sob a forma de *sentenças* que representam as crenças do agente sobre o mundo como ele é e seus objetivos modelados [44][2]. Essas sentenças são armazenadas em uma BC e representam o conhecimento do agente.

Uma linguagem de representação de conhecimento é definida por sua *sintaxe*, que especifica a estrutura de sentenças, e por sua *semântica*, que define o valor de verdade de cada sentença em um modelo de mundo possível. Esta linguagem pode ser representada por diversos formalismos lógicos, como por exemplo, lógica proposicional, lógica de primeira ordem, lógica descritiva, lógica modal, entre outros.

A sintaxe que exprime a idéia de consequência lógica entre sentenças na forma clausal é conhecida como cláusulas de Horn, as quais podem ser melhor entendidas em [43, 71]. Esta sintaxe expressa basicamente cláusulas na forma $B \leftarrow A$, onde B é a consequência lógica de A (relação causa-consequência), possibilitando facilmente representar o conhecimento do agente através de regras declarativas da forma *"se A, então B"*. Nesta representação, o lado antecedente da regra denota as condições necessárias para causar uma consequência, enquanto o lado consequente representa um conjunto de ações que serão tomadas de acordo com a causa satisfeita.

[2]Um agente lógico utiliza seus objetivos e crenças, expressos em forma de sentenças lógicas, para ter controle sobre seu comportamento [44].

Regras de inferência[3] são padrões de inferência consistentes que podem ser utilizadas em sentenças lógicas para descobrir provas [71]. O princípio da resolução introduz um mecanismo de inferência completo para sentenças lógicas na *forma normal conjuntiva (FNC)*[4]. Seguindo este princípio, para se provar a validade da sentença $\beta \leftarrow \alpha$, deve-se concluir por refutação ou contradição que sua equivalência lógica $\neg\beta \wedge \alpha$ na FNC seja não-satisfatível.

A inferência com cláusulas de Horn[5], seguindo o princípio da resolução, pode ser feita através dos seguintes algoritmos de refutação [44, 71]:

- **encadeamento progressivo**: raciocínio do antecedente para o consequente, partindo de um conjunto de fatos ou causas sobre o mundo, onde estes fatos implicam em consequências que podem derivar novas representações e consequências sobre o mundo, das quais sucedem-se até que um objetivo ou ação seja alcançado; e
- **encadeamento regressivo**: raciocínio do consequente para o antecedente, partindo de um conjunto de hipóteses (objetivos) sobre o mundo, onde estas hipóteses se reduzem a objetivos menores, dos quais sucedem-se na busca por evidências que sustentem suas hipóteses.

É útil visualizar encadeamento progressivo e regressivo em termos de caminhos em um espaço de busca do problema, onde estados intermediários correspondem às hipóteses intermediárias através do encadeamento regressivo, ou conclusões intermediárias através do encadeamento progressivo. Por encadeamento regressivo, o esclarecimento das ações é facilitado, uma vez que a busca parte de um objetivo já conhecido como hipótese, reduzindo-se a objetivos específicos, na busca por evidências que suportem suas hipóteses. Este comportamento exibe um raciocínio dirigido a objetivos. Em contrapartida, por encadeamento progressivo, o esclarecimento das ações não é facilitado, visto que a busca parte de fatos percebidos sobre o mundo e, a partir destes, derivam-se novos fatos, na tentativa de descobrir quais são seus objetivos e deduzir suas devidas ações. Este comportamento exibe um raciocínio dirigido a dados (fatos).

A Tabela 3.1 apresenta um resumo das principais características do encadeamento progressivo e regressivo.

[3] Regras de inferência não são regras declarativas que representam o conhecimento do mundo como ele é. São mecanismos utilizados para fazer asserções de senteças lógicas.

[4] A forma normal conjuntiva é uma conjunção de disjunções: $(a_{1,1} \vee \cdots \vee a_{1,n}) \wedge \cdots \wedge (a_{m,1} \vee \cdots \vee a_{m,n})$, equivalente à lógica proposicional. A regra de resolução se aplica apenas às disjunções de literais, como explicado em [71].

[5] A inferência com cláusulas de Horn é o mecanismo utilizado por motores de inferência baseados em regras, explicados em seguida na Seção 3.2.3.

Tabela 3.1: Principais características de encadeamento progressivo e regressivo [35].

Encadeamento progressivo	**Encadeamento regressivo**
Planejamento, monitoramento, controle	Diagnóstico
Presente para o futuro	Futuro para o passado
Antecedente para consequente	Consequente para antecedente
Dirigido a dados, raciocínio *bottom-up*	Dirigido a objetivos, raciocínio *top-down*
Avanço progressivo para encontrar quais soluções seguem a partir dos fatos	Retroage para encontrar fatos que suportam as hipóteses
Busca em largura facilitada	Busca em profundidade facilitada
Antecedentes determinam a busca	Consequentes determinam a busca
Esclarecimento não facilitado	Esclarecimento facilitado

3.2.3 Motores de inferência baseados em regras

O raciocínio lógico aplicado a uma sintaxe de regras declarativas, explicado na Seção 3.2.2, introduz mecanismos de raciocínio utilizado por motores inferência baseado em regras, que podem ser tanto por encadeamento progressivo quanto regressivo. Esta seção apresenta uma breve explicação dos componentes básicos de um motor de inferência baseado em regras e, em seguida, apresenta descrições de vários motores disponíveis gratuitamente.

Componentes básicos

Um motor de inferência de regras geralmente contém os seguintes componentes [37]:

- **casamento de padrões**: realiza o casamento (unificação) entre regras e fatos, onde as regras que são casadas com os fatos deverão ser ativadas para execução, devendo-se estabelecer uma ordem para solucionar possíveis conflitos de execução;
- **agenda**: gerencia a ordem de execução das regras ativas por meio de estratégias de resolução de conflitos, decidindo quais das regras ativas terão maior prioridade e deverão ser executadas primeiro; e
- **motor de execução**: executa as regras que estão ativas e ordenadas na agenda, podendo a execução ser tanto por encadeamento progressivo quanto regressivo, sendo que estes mecanismos de inferência podem ser combinados.

Ferramentas

Drools [65], desenvolvido na linguagem Java, é um motor de inferência baseado em regras que possui os mecanismos de inferência tanto por encadeamento progressivo quanto regressivo, sendo este último disponível a partir da versão 6.x. Seu motor de inferência é

implementado com os algoritmos Rete [28] e Leaps [14], possibilitando realizar *casamento de padrões* com eficiência entre fatos (novos e/ou existentes) e regras. O Drools pode ser usado em vários tipos de projetos, pois possibilita combinar a inferência de regras com processamento de eventos complexos, e também com *workflows* e processos de planejamento automatizado, classificando-o como um sistema de gerenciamento de regras de negócio (*Business Rule Management System BRMS*).

Jess [37], desenvolvido na linguagem Java, possui mecanismos de inferência de regras por encadeamento progressivo e regressivo. Assim como o Drools, Jess utiliza o algoritmo Rete para realizar casamento de padrões entre regras e fatos.

Apache Jena [1], escrito na linguagem Java, possui mecanismos de inferência voltadas para web semântica. Seu motor de inferência apresenta dois tipos de raciocínios: baseado em regras e baseado em ontologias. Os mecanismos de raciocínio baseado em regras incluem raciocínios por encadeamento progressivo e regressivo, podendo ser combinados para resolução das regras. Os mecanismos de inferência baseado em ontologias podem operar sobre as linguagens OWL[6] e RDF Schema[7]. Ontologias e regras podem ser combinadas em uma única BC, podendo seu motor de inferência realizar um mecanismo de inferência híbrido entre regras e ontologias.

JEOPS [23], implementado na linguagem Java, possui apenas o mecanismo de inferência por encadeamento progressivo. Também utiliza o algoritmo Rete para realizar casamento de padrões. Devido às suas capacidades, JEOPS é classificado como um sistema de regras de produção. Esse termo é utilizado em sistemas com mecanismos de encadeamento progressivo, onde a partir dos fatos iniciais, novos fatos serão produzidos, na busca de alcançar seus objetivos.

PROLOG [53] foi criado com o intuito de fornecer uma linguagem de programação em Lógica Matemática, baseada no conceito de resolução linear (SL-Resolution) proposto por Kowalski[8] [42, 45]. Possui apenas o mecanismo de inferência por encadeamento regressivo. Porém, pode ser utilizado para implementar sistemas especialistas, assim como agentes baseados em conhecimento.

A Tabela 3.2 apresenta um resumo das ferramentas descritas.

[6]Web Ontology Language (OWL): É uma linguagem para definir e instanciar ontologias na web, composta por uma diversidade de vocabulários que permitem expressar a semântica das coisas, incluindo vocabulários da linguagem RDF.

[7]Resource Description Framework Schema (RDF Schema): provê elementos básicos para descrição de ontologias.

[8]As contribuições de R. Kowalski foram utilizadas juntamente com as definições apontadas por Russell & Norvig para explicar os conceitos de agentes baseados em conhecimento apresentados nesta dissertação.

Tabela 3.2: Resumo dos motores de inferência.

Ferramenta	**mecanismo de inferência**
Drools	encadeamento progressivo e regressivo
Jess	encadeamento progressivo e regressivo
Jena	híbrido (inferência de regras e ontologias)
JEOPS	encadeamento progressivo
PROLOG	encadeamento regressivo

3.3 Especificações recomendadas para *frameworks* de SMA

A Fundação para Agentes Físicos Inteligentes (*Foundation for Intelligent Physical Agents - FIPA*) [32] é uma organização internacional da IEEE responsável por estabelecer normas e especificações que apoiam a interoperabilidade entre agentes e aplicações baseadas em agentes. Desde sua fundação, a FIPA tem desempenhado um papel importante no desenvolvimento de normas para agentes e tem promovido uma série de iniciativas e eventos que contribuíram para o desenvolvimento e utilização de suas especificações.

Suas especificações estão dividias em cinco eixos:

1. **comunicação de agentes**: estas especificações lidam com mensagens ACL (*Agent Communication Language* ou linguagem de comunicação do agente), protocolos de interação de troca de mensagens, atos de fala baseados em teorias de atos comunicativos e linguagens de representação de conteúdo, tais como performativas e ontologias;

2. **transporte de mensagens de agentes**: estas especificações lidam com o transporte e representação das mensagens através de diferentes protocolos de tranporte em ambientes de redes cabeadas ou sem fio;

3. **gerenciamento de agentes**: estas especificações lidam com o gerenciamento de agentes em plataformas de agentes e entre plataformas, tais como mobilidade de agentes, gerenciamento de serviços, ciclo de vida e entre outros;

4. **arquitetura abstrata**: estas especificações lidam com entidades abstratas que são necessárias para construção de serviços essenciais utilizados por agentes; e

5. **aplicações**: estas especificações incluem alguns exemplos de áreas de aplicação nos quais agentes FIPA podem ser implantados.

3.4 Ferramentas de SMA

Esta seção apresenta diferentes ferramentas para desenvolvimento de SMA, expondo suas principais características, que podem ou não estar de acordo com as especificações recomendadas pela FIPA, descritas na Seção 3.3.

Java Agent DEvelopment Framework (JADE) [15] é um *framework* de desenvolvimento de SMA, implementado na linguagem Java, que segue as especificações da FIPA. Sua arquitetura fornece interoperabilidade entre agentes, possibilitanto que os agentes de um SMA estejam dispostos em diferentes meios físicos, tais como smartfones, computadores e tablets. De acorodo com Wooldridge [88], JADE é a ferramenta pública mais conhecida na comunidade científica. Os mecanismos de comunicação entre agentes fornecidos no JADE inclui apenas troca de mensagens e seu *framework* não inclui componentes prontos para criação de agentes autônomos, por exemplo suporte à criação de um agente com arquitetura baseado no modelo *Beliefs, Desires and Intentions* (BDI). Cabe ao desenvolvedor projetar a arquitetura interna de cada agente.

JADE eXtension (Jadex) [63] é uma implementação de uma arquitetura de agente híbrida (reativa e deliberativa) para representar capacidades cognitivas em agentes JADE, seguindo o model BDI. Agentes Jadex mantém as mesmas funcionalidades de um agente JADE e são capazes de se comunicar com agentes JADE por meio de troca de mensagens. Cada agente Jadex requer um conjunto de descrições declarativas em um arquivo que representam suas crenças, objetivos e planos. Estas descrições especificam os mecanismos de raciocínio do agente.

Cougaar [80] é uma arquitetura implementada na linguagem Java para construção de aplicações distribuídas baseadas em agentes. Sua arquitetura não está de acordo com as especificações da FIPA. Todavia, fornece um mecanismo híbrido de comunicação, podendo agentes trocar informações via troca de mensagens e através de uma memória compartilhada denominada *blackboard*. Cada agente contém módulos pré-definidos pronto para uso e construção de sistemas multiagentes cognitivos, classificando-os como agentes de software. Tal modularidade fornece a capacidade de desenvolver agentes com alta capacidade de autonomia, porém torna o desenvolvimento inflexível, impossibilitando o desenvolvimento de agentes que requer uma arquitetura diferenciada do modelo proposto.

MadKit [36] é uma plataforma multiagente modular e escalável, escrita na linguagem Java, que permite a criação de SMA baseado no modelo organizacional *agente, grupo e papel*. Seus agentes são entidades de comunicação ativa que desempenham papéis em grupos no qual participam. Cada agente pode criar ou participar de um ou mais grupos e, em cada grupo, pode desempenhar vários papéis. No momento, sua plataforma não está de acordo com as especificações da FIPA.

ZEUS [22] oferece um conjunto de componentes e utilidades, implementados na linguagem Java, que dão suporte ao desenvolvimento de SMA, de acordo com as especificações da FIPA. Dentre eles, inclui ferramentas que auxiliam a edição de agentes com ferramentas visuais, tais como editor de ontologias, ferramentas de gerencimanto e geração de código. Como mecanismos de raciocínio e coordenação, ZEUS fornece um sistema de escalonamento e planejamento adequado para aplicações orientadas a tarefas.

A Tabela 3.3 apresenta um resumo das ferramentas citadas acima.

Tabela 3.3: Resumo dos *frameworks* de SMA.

Framework	**Conforme com especificações FIPA**	**Principais características**
JADE	sim	SMA distribuído, modelagem simples
Jadex	sim	modelagem BDI
Cougaar	não	Memória operacional compartilhada (*Blackboard*)
MadKit	não	modelagem organizacional *agente, grupo e papel*
ZEUS	sim	modelagem de aplicações orientadas a tarefas

3.5 Discussão

O conceito de agentes inteligentes, explicado na Seção 3.1, é coberto por diversas definições, não havendo uma definição única e geral que conceitue um agente inteligente. Dentre as definições, um agente inteligente é capaz de assumir controle sobre seu comportamento, geralmente por meio de uma medida de performance que o auxilie na tomada de suas decisões, o que lhe garante apresentar um certo grau de autonomia.

Em vista de uma conceituação ampla, diversas metodologias com fundamentos bastante diferentes foram e estão sendo desenvolvidas na busca de criar agentes autônomos, considerando as arquiteturas no micro-nível (arquitetura interna de agentes) e no macro-nível (interação social de agentes). Num nível micro, foram destacados neste capítulo uma arquitetura geral do agente e uma arquitetura de agente baseado em conhecimento, que é a base de entendimento dos agentes propostos nesta dissertação. No nível macro, foram descritas duas capacidades básicas de interação: competição e colaboração.

As ferramentas de SMA geralmente incluem componentes para desenvolvimento nos níveis micro e macro. O Jadex é um exemplo de ferramenta que inclui componentes no nível micro prontos para desenvolvimento de agentes com a arquitetura BDI. Por outro lado, as ferramentas MadKit, Cougaar e ZEUS preocupam-se em fornecer componentes específi-

cos para desenvolvimento de uma comunicação inteligente no nível macro. Diferentemente destas ferramentas, o JADE inclui apenas os componentes básicos de desenvolvimento de SMA, cabendo ao desenvolvedor implementar seus próprios mecanismos nos níveis micro e macro.

A vantagem das ferramentas que incluem modelos de desenvolvimento específicos é a facilidade de criação de agentes com boa capacidade de autonomia. Por outro lado, desenvolvedores não tem tanta flexibilidade de projetar um sistema com caracteristicas muito diferentes dos componentes fornecidos pelas ferramentas. Em relação ao levantamento de ferramentas, apresentado na Seção 3.4 deste Capítulo, o JADE é um *framework* simples que fornece os componentes básicos para desenvolvimento de SMA. Por sua simplicidade, o JADE fornece maior flexibilidade ao desenvolvedor para projetar agentes com diferentes arquiteturas em um ambiente multiagente.

Capítulo 4

BioAgents-Prot

O BioAgents-Prot foi desenvolvido para simular a anotação manual de proteínas em projetos transcritoma, um processo em que o conhecimento biológico e a experiência de anotação são utilizados para atribuir funções biológicas ou encontrar características genômicas das sequências, previamente anotadas por programas e bancos de dados com sequências anotadas.

A simulação deste processo no BioAgents-Prot é realizada pela análise e interpretação dos resultados produzidos pela fase de anotação automática, de acordo com o conhecimento armazenado no BioAgents-Prot. Deve-se notar que, com regras simples e poucas ferramentas conseguimos obter bons resultados, o que indicar que, com um conhecimento mais especializado, poderíamos obter uma anotação ainda mais acurada.

Na Seção 4.1, descrevemos a arquitetura proposta para o BioAgents-Prot, enquanto na Seção 4.2, detalhamos o protótipo implementado.

4.1 Arquitetura

O conhecimento biológico das proteínas, utilizado no processo de anotação, foi extraído e formalizado em uma representação de regras claras e bem definidas, com o apoio dos biólogos. Essa representação foi dividida em três diferentes abordagens:

1. similaridade de sequências (inferência de função por “homologia”);
2. característica de domínios conservados; e
3. conservação em espécies relacionadas.

Dessa forma, a arquitetura do BioAgents foi dividida em três camadas: interface, colaborativa e física, como apresentada na Figura 4.1. O conhecimento utilizado na camada

colaborativa é distribuído entre agentes que trabalham em conjunto, com o objetivo de obter anotações confiáveis.

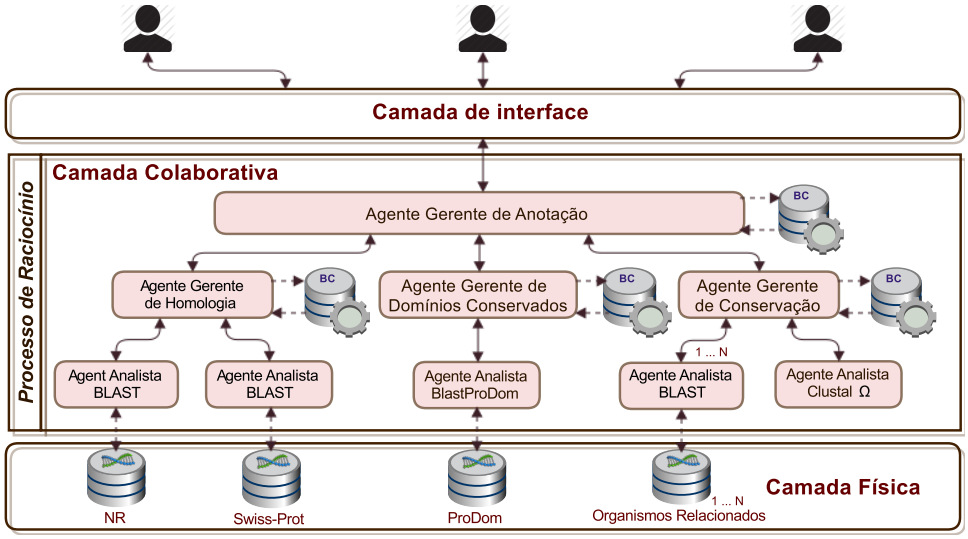


Figura 4.1: A arquitetura de três níveis do BioAgents-Prot.

A camada de interface modela a interface entre o usuário e o núcleo do sistema através de uma interface web.

A camada colaborativa é o núcleo da arquitetura, composta por diferentes agentes gerentes (GR) e analistas (ANL), onde cada ANL possui um *parser* para analisar os resultados produzidos por ferramentas específicas. Por outro lado, cada GR possui uma base de conhecimento BC associada e um mecanismo de inferência que permite raciocinar sobre os resultados produzidos por seus ANLs e outros GRs. Modelados pelo conhecimento extraído da biologia, com auxílio de biólogos, os agentes gerentes trabalham colaborativamente na tentativa de alcançar uma anotação de proteína confiável.

A camada física é formada por diferntes bancos de dados disponíveis publicamente, por exemplo NR, Swiss-Prot e ProDom.

Uma descrição mais detalhada de cada camada da arquitetura é apresentada a seguir.

Camada de interface

A camada de interface enfilera e escalona as requisições do usuário utilizando a política *First Come First Serve* (FCFS), retornando os resultados para o usuário através de uma interface web. A requisição do usuário consiste em uma lista de sequências (no formato FASTA) para ser anotado pelo BioAgents-Prot, além dos parâmetros das ferramentas e seleção de bancos de dados. Os resultados produzidos podem ser exportados em quatro formatos: PDF, XLS, CSV e XML.

Camada colaborativa

Na camada colaborativa, o agente gerente de anotação (GRA) coordena diferentes GRs, cada um realizando uma tarefa de anotação particular. A atual versão do BioAgents-Prot possui três GRs com diferentes abordagens de anotação, baseadas em: homologia, características de domínios conservados (como um refinamento para a anotação por homologia) e conservação em espécies relacionadas. Cada GR coordena um grupo de ANLs para trabalharem em conjunto a fim de alcançarem a melhor sugestão de anotação, referente à uma abordagem particular. Estes agentes são descritos a seguir:

- GRA define um *pipeline* de anotação dinâmica alocando diferentes GRs para anotarem uma sequência. Dependendo dos resultados que vem sendo produzidos por GRs, o GRA irá determinar que outro GR poderá refinar ou sugerir uma nova anotação, colocando os agentes para trabalharem de maneira colaborativa a fim de alcançarem uma anotação confiável. Dentre as possibilidades, o GRA busca sugerir a anotação mais confiável;
- GRs comportam-se de maneira análoga ao GRA. Estes agentes definem um meta-*pipeline*, decidindo quais ANLs deverão executar e extrair informações novas e úteis para o sistema. O agente gerente de homologia (GRH) raciocina sobre os resultados dos ANLs e decide que anotação será sugerida ao GRA. O agente gerente de domínios conservados (GRDC) busca refinar a sugestão do GRH, comparando (textualmente) a sugestão do GRH com descrições funcionais recuperadas dos alinhamentos das sequências de domínios conservados, armazenadas no banco ProDom. O agente gerente de conservação (GRC) irá buscar por conservação em bancos de espécies relacionadas para sugerir anotação como *"conserved hypothetical protein"*.
- Cada ANL realiza um *parser* dos resultados produzidos por ferramentas particulares (por exemplo, BLAST, Clustal Ω e BastProDom) para recuperar e enviar informações úteis ao seu correspondente GR.

Camada física

A camada física contém os seguintes bancos de dados biológicos: NR [2], SwissProt [19], ProDom [74] e bancos de espécies relacionadas, utilizados apropriadamente pelos ANLs.

Anotação por homologia

Como apresentado na Figura 4.1, os agentes no BioAgents-Prot estão agrupados em três diferentes tipos de conhecimento em homologia, que foram baseadas em: similaridade de sequências, domínios conservados e conservação com espécies relacionadas. O conhecimento dessas tarefas corresponde às bases de conhecimento dos GRs, definidos para realizar nosso estudo de caso, mas outros tipos específicos de conhecimento, ferramentas e bancos de dados também poderiam ser adotadas, de acordo com as características do projeto.

4.2 Protótipo

A atual abordagem do BioAgents-Prot combina três diferentes abordagens (como descritas anteriormente) para atribuir uma anotação às sequências de proteínas. BioAgents-Prot foi implementado com o *framework* JADE para o desenvolvimento de SMA, enquanto o raciocínio de anotação de proteínas foi definido para cada GR com o motor de inferência Drools.

JADE foi utilizado por uma série de razões. O JADE é um software livre, distribuído pela licença *LGPL*. Suas especificações são compatíveis com os padrões definidos pela FIPA e, em comparação com as ferramentas apresentadas na Seção 3.4, discutidas em seguida na Seção 3.5, o JADE oferece componentes que garantem maior flexibilidade em relação às outras ferramentas. Além disso, pode-se perceber que há uma comunidade ativa de usuários e desenvolvedores, que produzem continuamente várias documentações.

Dentre os vários motores de inferência descritos na Seção 3.2.3, o Drools tem a comunidade mais ativa. Seus projetos derivados [65], tais como OptaPlanner, jBPM e Fusion, podem ser integrados permitindo construir mecanismos de inferências complexos. Sua linguagem é flexível o suficiente para combinar a semântica de um domínio do problema em particular com linguages específicas de domínio (DSL), permitindo estabelecer uma linguagem legível para o especialista. Portanto, o Drools foi utilizado como motor de inferência de regras dos GRs e as regras declaradas com o Drools, em cada GR, representam o conhecimento biológico utilizado no processo de anotação.

Os *parsers* dos ANLs, utilizados para extrair informações úteis de resultados de várias ferramentas, foram implementado usando expressões regulares na linguagem Java. Estes suportam os formatos XML, texto e HTML.

Deve-se notar ainda que a versão mais recente do ProDom [74] é do ano de 2010. Desde então, não houve mais atualizações desse banco. Desde a versão de 2006, não foram disponibilizados o arquivos no formato FASTA das famílias de domínios. Estes arquivos foram

disponibilizados até a versão de 2005, que são: (i) arquivo de regiões conservadas provenientes do alinhamento múltiplo de cada família; e (ii) arquivo do consenso das regiões conservadas de cada família. Entretanto, foi disponibilizado um banco de indexação no formato SRS utilizado para consultas posteriores, que contém todas estas informações. Este formato SRS pode ser melhor entendido em [31].

O script BlastProDom [74] requer um banco indexado para ser utilizado com a ferramenta BLAST. Como consequência, foi necessário fazer uma engenharia reversa para reconstruir as sequências de alinhamentos mútiplos e de consenso, analisando as famílias contidas no banco SRS e as sequências geradas contidas na versão de 2005. Um *script* na linguagem Perl foi desenvolvido para geração destas sequências e pode ser consultado no Anexo A. A partir dessas sequências, foi possível reconstruir o mesmo banco de 2010, disponível para consultas no *site* do ProDom. Além disso, o BlastProDom foi modificado para utilizar a versão mais recente do BLAST. Algumas correções de código também foram necessárias, para que o *script* funcionasse corretamente.

4.2.1 Descrição dos agentes

O GRA primeiramente recebe uma requisição, enviada por um usuário, e a encaminha para o GRH, buscando inicialmente obter uma recomendação de anotação por homologia. Se isso não for possível, o GRA encaminha a requisição para o GRC verificar se há conservação.

Como discutido anteriormente, como a similaridade funcional apresenta uma fraca correlação com a similaridade de sequências, GRA combina os resultados dos agentes GRH e GRDC, calculando um escore de predição para ambos os métodos [68]. O GRA considera a anotação do GRH *confiável* se o GRDC confirma esta anotação (com alinhamentos de domínios conservados compatíveis), caso contrário, o GRA analisará se o alinhamento da anotação recomendada pelo GRH é suficiente (porém não confiável) para recomendar anotação. O GRA possui três conjuntos de regras, definidos para tratar os resultados obtidos de cada GR. Nestes conjuntos de regras, os escores fornecidos pelos gerentes GRDC e GRC serão descritos em seguida. Os conjuntos de regras são descritos como segue:

- conjunto de regras para tratar resultados do GRH:
  - se GRH sugere anotação α, então aloca GRDC para confirmá-la com domínios conservados;
  - se GRH não sugere anotação, então aloca GRC para verificar por conservação com espécies relacionadas.

- conjunto de regras para tratar os resultados do GRDC:
  - se GRDC confirma a anotação α do GRH com *escore* $\geq$ 70%, então reporta para o usuário uma anotação confiável;
  - se GRDC confirma a anotação α do GRH com escore entre 50% e 70%, então α é reportado como uma anotação suficiente;
  - se GRDC não confirma a anotação α do GRH (escore $\leq$ 50%) e:
    - se há no mínimo uma sugestão do GRH com identidade $\geq$ 70% e positivo $\geq$ 80%, então reporta como anotação inferida por similaridade;
    - se não há sugestões do GRH com identidade $\geq$ 70% e positivo $\geq$ 80%, então aloca GRC para verificar por conservação em organismos relacionados.
- conjunto de regras para tratar os resultados do GRC:
  - se GRC reporta conservação com escore $\geq$ 50%, então GRA reporta ao usuário *"conserved hypothetical protein"* com boa conservação entre espécies;
  - se GRC reporta conservação com escore entre 30% e 40%, então GRA reporta ao usuário *"conserved hypothetical protein"* com fraca conservação entre espécies;
  - se GRC reporta conservação $\leq$ 30% ou não encontra nenhum tipo de conservação, então GRC reporta ao usuáro que nenhuma sugestão foi encontrada.

O *workflow* apresentado na Figura 4.2 ilustra o funcionamento e raciocínio do GRA.

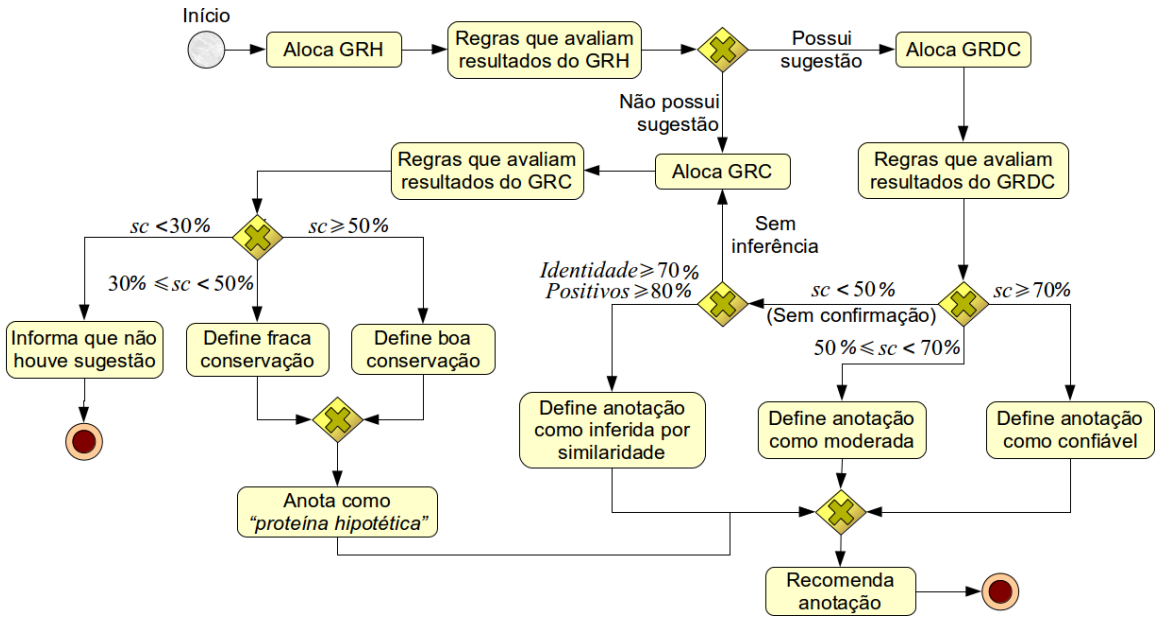


Figura 4.2: Processo de raciocínio do GRA em notação BPMN.

GRH aloca dois ANLs para que executem a ferramenta BLAST em paralelo com os bancos NR e Swiss-Prot e, em seguida, realizam o *parser* de seus resultados. Esses ANLs consultam os *hits* dos alinhamentos (acima de um *threshold* fornecido como parâmetro do sistema), na tentativa de encontrar uma anotação diferente de *"hypothetical"*, *"putative"* e outros termos relacionados, ou seja, busca por anotações que possuam uma descrição de função. Se o ANL tiver sucesso na busca, informa ao GRH o *hit* que possui descrição funcional com maior escore. Caso contrário, se entre as anotações não for encontrado uma descrição funcional, é informado ao GRH apenas o melhor *hit*. Em seguida, o GRH irá decidir qual das anotações recuperadas por ambos analistas será informado ao GRA, de acordo com as regras descritas mais tarde. O conjunto de regras do GRH é executado em ordem de prioridade[1] (quanto maior é o número de prioridade associada a uma regra, maior será sua prioridade). Estas regras são descritas a seguir:

- prioridade 2: se há um *hit* do Swiss-Prot diferente de *"hypothetical"*, então sugere esta anotação ao GRA (Código 4.1);
- prioridade 1: se há um *hit* do NR diferente de *"hypothetical"*, então sugere esta anotação ao GRA;

O *workflow* apresentado na Figura 4.3 ilustra o funcionamento e raciocínio do GRH.

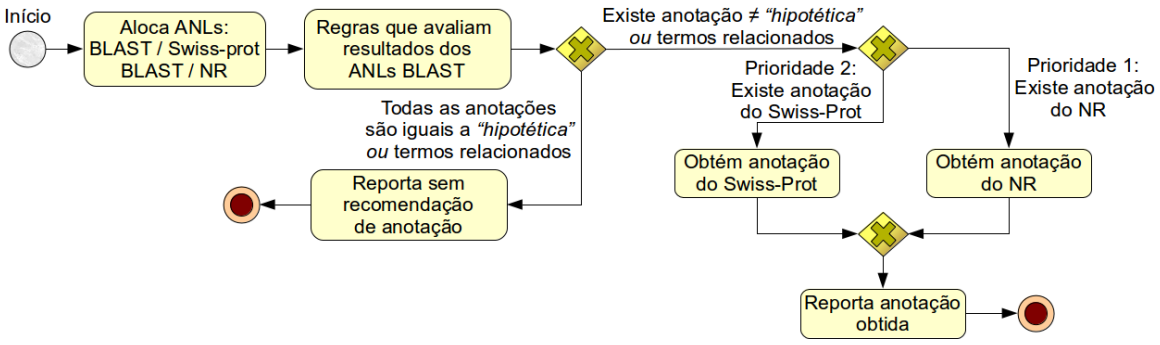


Figura 4.3: Processo de raciocínio do GRH em notação BPMN.

GRDC aloca um ANL para executar o *script* BlastProDom e informar ao GRDC uma lista contendo o melhor *hit* de cada família de domínios. Uma vez que ProDom foi criado utilizando sequências de proteínas do Swiss-Prot (que possui anotações funcionais curadas), é possível computar textualmente um escore, comparando a anotação do GRH

[1]Prioridades podem ser estabelecidas no Drools utilizando o termo *salience* (por exemplo, uma regra definida com *salience 5* tem maior prioridade que uma regra definida com *salience 2*). *Salience* é utilizado nas regras do Drools como uma estratégia de resolução de conflitos para decidir que regra deverá ser executada primeiro (veja Seção 3.2.3).

com as descrições fornecidas por cada família. Este escore (sc) é calculado como:

$$sc = \frac{\sum_{i=1}^{|D|} M(d_i, \alpha)}{|D|} \tag{4.1}$$

Onde, $D = \{d_1, d_2, \ldots, d_n\}$, $|D| = n$, é um conjunto de elementos de famílias de domínios com n anotações; d_i, tal que $i = 1, \ldots, n$, representa a anotação de cada família; α é a anotação funcional sugerida por GRH; e M é a função de casamento entre anotações d_i e α, que retorna 1 se as anotações casam ou 0, caso contrário.

Uma vez que estamos computando textualmente o casamento entre anotações, as anotações atribuídas às famílias como *"hypothetical"* e termos relacionados são desconsideradas de D. Foi observado que o ProDom (2010) possui 951.264 famílias com anotações irrelevantes, de um total de 2.749.601 famílias. De 951.264 famílias, 755.483 são formadas por subsequências de uma única espécie, que não são úteis em nosso contexto, sendo as demais famílias formadas por subsequências de duas ou mais espécies. Estas famílias foram identificadas com um filtro de palavras irrelevantes.

O conjunto de regras do GRDC é descrito a seguir:

- se $sc \geq 70\%$, então reporta ao GRA a anotação do GRH (se este for o caso) como **confiável** (Código 4.2);
- se $50\% \leq sc < 70\%$, então reporta ao GRA a anotação do GRH como uma anotação possível;
- se $sc < 50\%$, então reporta ao GRA que não houve confirmação.

O *workflow* apresentado na Figura 4.4 ilustra o funcionamento e raciocínio do GRDC.

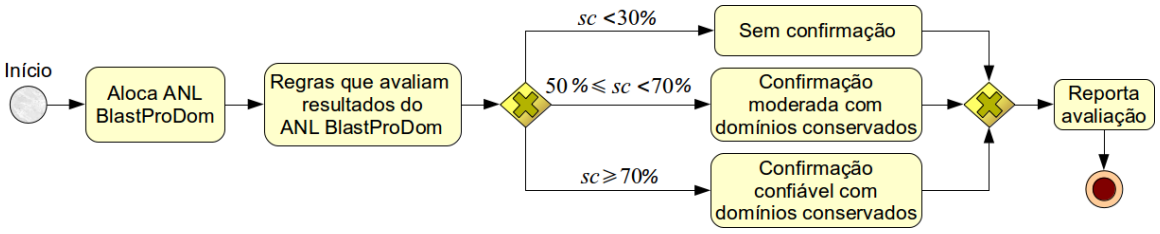


Figura 4.4: Processo de raciocínio do GRDC em notação BPMN.

GRC possui dois conjuntos de regras utilizados para tratar os resultados de dois tipos diferentes de ANLs: um executa a ferramenta BLAST, buscando pela sequência de melhor *hit*; e outro executa a ferramenta Clustal Ω (utilizada para realizar um alinhamento múltiplo de sequências) e calcula a conservação do resultado produzido. Esse agente, de forma análoga ao GRA, define um *pipeline* de acordo com as condições atuais do sistema. Primeiramente, o GRC aloca m ANLs para executar em paralelo a ferramenta BLAST

com m bancos de espécies relacionadas. Cada ANL informa o melhor *hit* para o GRC, juntamente com sua sequência correspondente (utilizando a ferramenta *blastdbcmd* [11]). Dependendo dos resultados obtidos desses ANLs, o GRC decidirá se deverá alocar (ou não) o ANL Clustal Ω para realizar alinhamento múltiplo de sequências. Se o Clustal Ω for alocado, este realiza um *parser* do resultado produzido, medindo a conservação do alinhamento múltiplo, que inclui todas as sequências de entrada. Esta medida é calculada conforme apresentado na Equação 4.2:

$$sc = \frac{\sum positivos}{|consenso|} \tag{4.2}$$

onde, os positivos, no consenso obtido pelo Clustal Ω, são representados pelos símbolos "." , ":" e "*", atribuídos a cada coluna do consenso do alinhamento múltiplo. O tamanho da cobertura corresponde ao comprimento total do alinhamento múltiplo, que inclui os positivos e os *gaps*.

As regras definidas no tutorial sobre o ClustalW da SWBIC[2] [3], utilizadas para avaliar estas medidas, foram adaptadas. As regras utilizadas pelo GRC são descritas como segue:

- conjunto de regras para resultados de todos os ANLs BLAST:
  - se há no mínimo 2 sequências de m bancos de dados diferentes, então GRC envia essas sequências, juntamente com a sequência fornecida pelo usuário, para o ANL Clustal Ω (realizar o alinhamento múltiplo e calcular sua conservação);
  - se não há no mínimo 2 sequências de m bancos de dados diferentes, então reporta ao GRA *"nenhuma conservação entre espécies relacionadas"* (Código 4.3).
- conjunto de regras para o resultado obtido do ANL Clustal Ω:
  - se há um grande número de simlaridades ($sc \geq 70\%$), então reporta ao GRA *"conservação entre espécies relacionadas"*;
  - se há um bom número de simlaridades ($50\% \leq sc < 70\%$), então reporta ao GRA *"sequências tendem a compartilhar funções"*;
  - se há um número considerável de similaridades ($30\% \leq sc < 50\%$), então reporta ao GRA *"sequências apresentam similaridades"*;
  - se há um baixo número de similaridades ($sc < 30\%$), então reporta ao GRA *"nenhuma conservação entre espécies relacionadas"*.

O *workflow* apresentado na Figura 4.5 ilustra o funcionamento e raciocínio do GRC.

[2]The Southwest Biotechnology and Informatics Center - SWBIC.

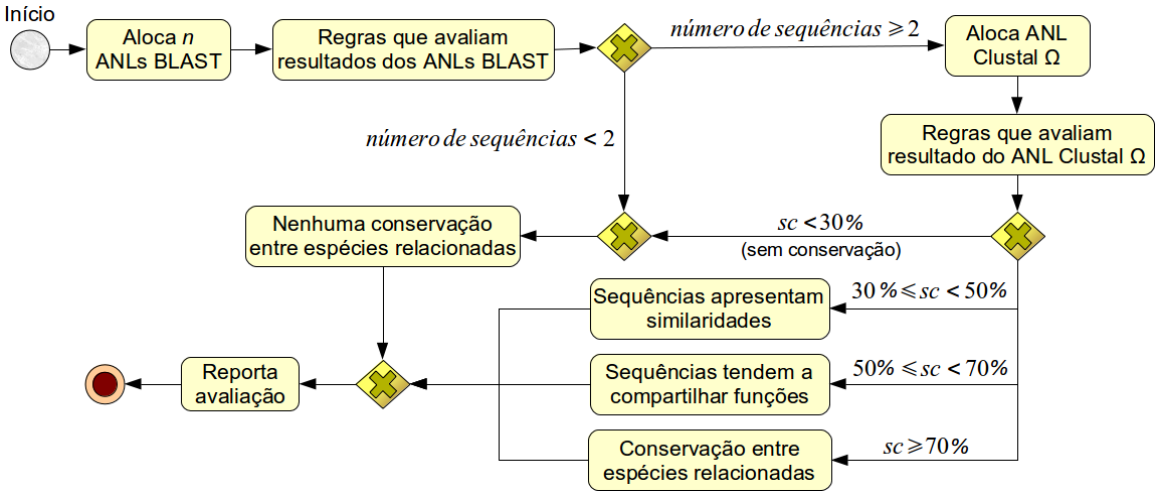


Figura 4.5: Processo de raciocínio do GRC em notação BPMN.

Exemplos de regras, apresentadas com sintaxe da linguagem Drools

As regras apresentadas a seguir são ilustradas com a sintaxe da linguagem Drools, utilizadas pelos GRs no BioAgents-Prot.

Código 4.1: Exemplo de regra utilizada pelo GRH.

```
rule "report Swiss-Prot annotation if its annotation != hypothetical"
salience 2
  when
    BlastHit ( db == "swissprot", $db : db,
      $desc : desc, !isHypothetical( desc ) )
  then
    suggestedDescription.append( $desc );
    suggestedDatabase.append( $db );
end
```

Código 4.2: Exemplo de regra utilizada pelo GRDC.

```
rule "Report as reliable annotation if score >= 70%"
  when
    BlastHit( $desc : desc )
    $pdSuggestion : ProdomSuggestion( $hitList : hits )
    $value : Double( this >= 0.7d ) from
      calculateScore( $desc, $hitList )
  then
    $pdSuggestion.setScore( $value );
    $pdSuggestion.setReport( "Reliable annotation [score (" +
      $value+  ") >= 70%]" );
end
```

Código 4.3: Exemplo de regra utilizada pelo GRC.

```
rule "Report no conservation between related species"
lock-on-active
agenda-group "ruleset for blast analyst results"
  when
    java.util.ArrayList ( size < 2 ) from
    collect ( AnalystSuggestion ( hit.getDesc() != "No hits found" ) )
  then
    stateBehaviour.setState(BehaviourStates.SEND_SUGGESTION);
    report.append("no conservation between related species");
end
```

4.2.2 Interface BioRequest e simulação do BioAgents-Prot

No BioAgents-Prot, a interface com o usuário foi feita através de um projeto Web, denominado BioRequest (Figura 4.6). Este projeto apresenta uma interface intuitiva, onde o biólogo pode submeter suas sequências e definir os parâmetros de execução do sistema. Através de mecanismos de validação, o BioRequest não permite submissão de parâmetros inválidos e, se este for o caso, exibirá mensagens destacadas de vermelho ao longo do formulário, como ilustradas na Figura 4.6. Estas mensagens são exibidas (i) localmente, próxima ao campo inválido; e (ii) globalmente, exibindo mensagens de todos os campos inválidos no cabeçalho do formulário.

Na atual versão do BioRequest, as requisições dos usuários são enfileiradas para execução, em ordem de chegada. Como consequência, uma requisição que está em espera deverá aguardar a execução de todas as outras que estão a sua frente. Quando uma requisição é finalizada, o usuário será redirecionado para a página de resultados, como ilustrada na Figura 4.7.

Os resultados exibidos para o usuário são apresentados em uma tabela, na qual os principais campos são: nome da sequência de entrada, sugestão de anotação e qualidade de anotação (Figura 4.7). A tabela completa dos resultados pode ser exportada para os formatos PDF, XML, XLS e CSV. Nessa tabela, uma interface de paginação é incluída, permitindo que o usuário possa navegar entre os resultados da tabela e, ao selecionar um resultado, uma informação detalhada de resultados e procedimentos de raciocínio do BioAgents-Prot é exibida. Deve-se notar que a qualidade de anotação fornecida é computada de acordo com as regras exibidas neste capítulo, a partir das regras incluídas nos agentes do BioAgents-Prot para inferir anotação.

Na Figura 4.8, são apresentados os resultados detalhados do transcrito SCRT_00012, listado na quinta linha da tabela apresentada na Figura 4.7. A recomendação final do BioAgents-Prot para este transcrito é que sua anotação é “confiável”, visto que a anota-

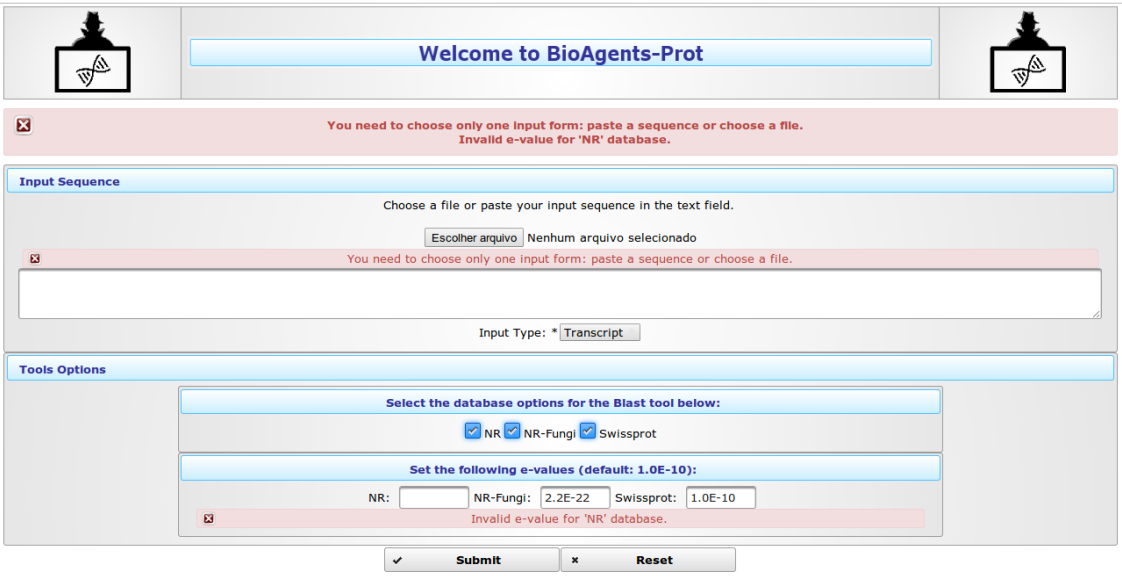
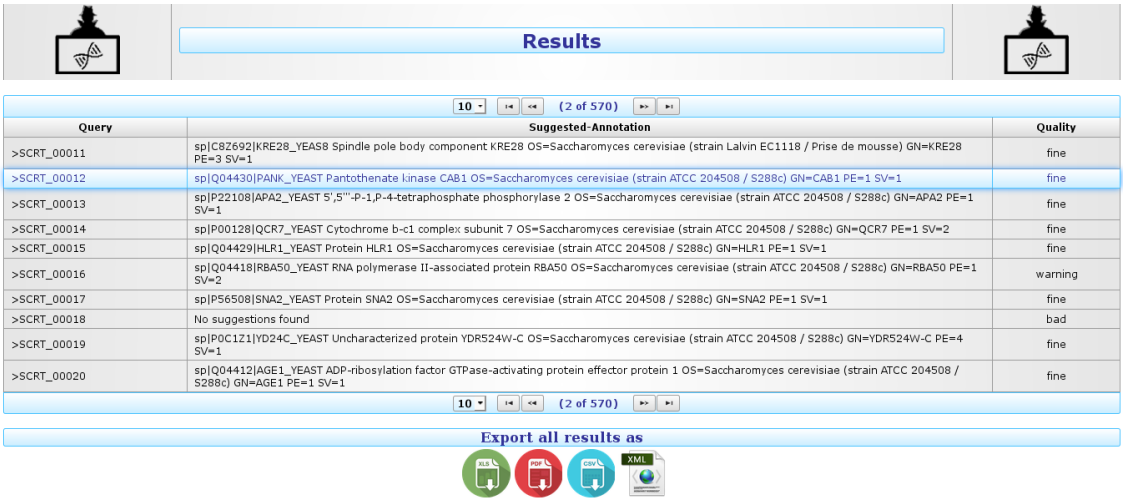


Figura 4.6: Página principal da interface BioRequest.



Query	Suggested-Annotation	Quality
>SCRT_00011	sp|C8Z692|KRE28_YEAS8 Spindle pole body component KRE28 OS=Saccharomyces cerevisiae (strain Lalvin EC1118 / Prise de mousse) GN=KRE28 PE=3 SV=1	fine
>SCRT_00012	sp|Q04430|PANK_YEAST Pantothenate kinase CAB1 OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=CAB1 PE=1 SV=1	fine
>SCRT_00013	sp|P22108|APA2_YEAST 5',5'''-P-1,P-4-tetraphosphate phosphorylase 2 OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=APA2 PE=1 SV=1	fine
>SCRT_00014	sp|P00128|QCR7_YEAST Cytochrome b-c1 complex subunit 7 OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=QCR7 PE=1 SV=2	fine
>SCRT_00015	sp|Q04429|HLR1_YEAST Protein HLR1 OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=HLR1 PE=1 SV=1	fine
>SCRT_00016	sp|Q04418|RBA50_YEAST RNA polymerase II-associated protein RBA50 OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=RBA50 PE=1 SV=2	warning
>SCRT_00017	sp|P56508|SNA2_YEAST Protein SNA2 OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=SNA2 PE=1 SV=1	fine
>SCRT_00018	No suggestions found	bad
>SCRT_00019	sp|P0C1Z1|YD24C_YEAST Uncharacterized protein YDR524W-C OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=YDR524W-C PE=4 SV=1	fine
>SCRT_00020	sp|Q04412|AGE1_YEAST ADP-ribosylation factor GTPase-activating protein effector protein 1 OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=AGE1 PE=1 SV=1	fine

Figura 4.7: Anotação dos transcritos da *S. cerevisiae*.

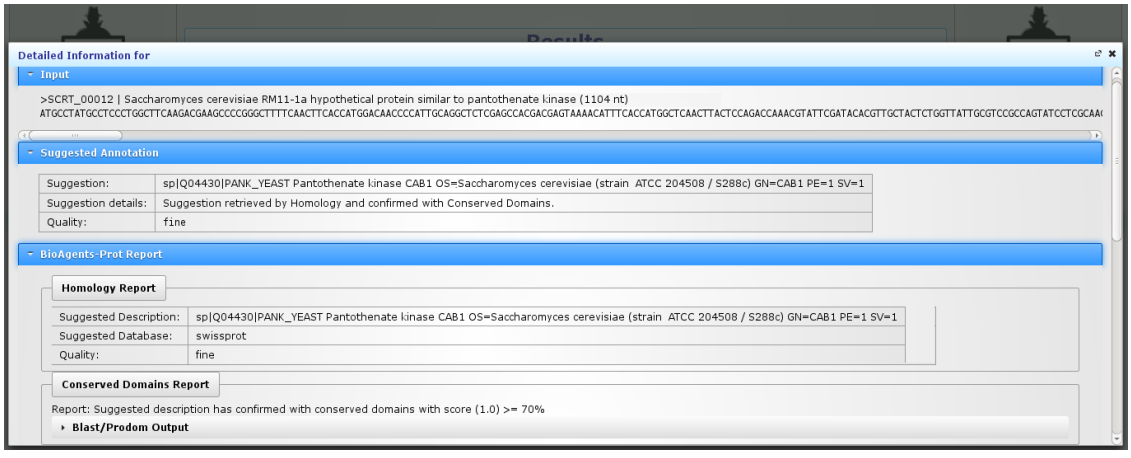


Figura 4.8: Resultados detalhados do transcrito SCRT_00012.

ção sugerida por homologia foi confirmada com domínos conservados (escore em 100%). Observamos que as informações detalhadas incluem todos os resultados utilizados pelas ferramentas no processo de simulação do BioAgents-Prot, podendo ser utilizadas para avaliação manual das anotações sugeridas pelo sistema.

Capítulo 5

Fungo *Saccharomyces cerevisiae*: um estudo de caso

Para validar BioAgents-Prot, foi realizado experimentos com o fungo *Saccharomyces cerevisiae*, um organismo modelo muito importante para pesquisas em genética e biologia de eucariotos. Como o primeiro genoma de eucarioto sequenciado, tornou-se também o modelo escolhido para a genômica funcional e comparativa, o que justifica seu uso como nosso padrão ouro.

Na Seção 5.1, descrevemos as características do fungo *Saccharomyces cerevisiae*. Na Seção 5.2, descrevemos como foram obtidos os dados e selecionados os parâmetros. Na Seção 5.3, mostramos como foram realizados os cálculos de similaridade funcional entre anotações. Na Seção 5.4, apresentamos as métricas utilizadas para avaliar o desempenho do BioAgents-Prot. Finalmente, na Seção 5.5, discutimos os resultados obtidos da comparação das anotações sugeridas pelo BioAgents-Prot e as anotações conhecidas da *S. cerevisiae*.

5.1 Descrição da *Saccharomyces cerevisiae*

Saccharomyces é um gênero no reino dos fungos que inclui muitas espécias de levedura, dentre as quais a *Saccharomyces cerevisiae* (Figura 5.1).

A *S. cerevisiae* é uma levedura bastante útil, sendo utilizada para a produção do pão, da cerveja, além do etanol. A *S. cerevisiae* é um dos organismos modelos de eucariotos mais intensamente estudados em Biologia Molecular, pois apresenta vantagens como: tamanho, tempo de geração, acessibilidade, manipulação, genética, conservação de mecanismos e potenciais benefícios econômicos.

A *S. cerevisiae* é um organismo modelo devido às seguintes características: é um organismo unicelular; divide-se por meiose, sendo um candidato para pesquisa em genética

Figura 5.1: Fungo *Saccharomyces cerevisiae* [6].

sexual; pode ser transformada por inclusão ou remoção de genes por recombinação homóloga; como é um eucarioto, tem a estrutura celular complexa de plantas e animais sem a quantidade de RNAs não-codificadores de eucariotos mais complexos; é economicamente importante e amplamente utilizada na indústria.

5.2 Dados e parâmetros selecionados

Os dados de transctritos [8] e de RNAs não codificadores (*non-coding RNAs* - ncRNAs) [9] da *S. cerevisiae* foram obtidos do *Instituto Broad* e do *Yeast Genome*, respectivamente. Estes transcritos, anotados manualmente como proteínas, e as sequências de ncRNAs nos permitem comparar as anotações sugeridas pelo BioAgents-Prot com ambos os conjuntos.

Para realizar nosso estudo de caso, foi construído o banco NR-Fungi (um conjunto pequeno do banco NR), contendo apenas sequências de fungos. As sequências de transcritos de organismos próximos filogeneticamente à *S. cerevisiae* foram obtidas do *Instituto Broad*, de acordo com a árvore filogenética apresentada no site. Um banco BLAST de sequências indexadas foi construído para cada um dos organismos obtidos. Como os transcritos das demais espécies da *Saccharomyces* não estão disponíveis no *Instituto Broad*, foi utilizado transcritos de 7 espécies da *Candida* [4] e 4 espécies da *Schizosaccharomyces* [5], um total de 11 bancos BLAST de diferentes espécies.

Os parâmetros de execução, definidos para submissão das sequências de transcritos, foram: $e-value \leq 10^{-10}$ para execução da ferramenta BLAST com os bancos NR-Fungi e Swiss-Prot e de espécies relacionadas. As demais ferramentas utilizaram seus parâmetros padrões. Para as sequências de ncRNAs, os parâmetros diferenciam-se apenas no $e-value$, sendo esse $\leq 10^{-5}$, uma vez que essas sequências possuem um tamanho (em nucleotídeos) pequeno.

5.3 Cálculo de similaridade funcional entre anotações

Uma vez que a performance do BioAgents-Prot foi medida comparando-se as anotações manuais do *Instituto Broad* com as anotações do BioAgents-Prot, foi adotado um cálculo que permitisse mensurar a similaridade funcional entre duas anotações.

A comparação entre estas anotações foi realizada a partir de um coeficiente de similaridade que determina o quanto as anotações são iguais. Assim, seja $A = \{a_1, a_2, \ldots, a_n\}$ um conjunto de palavras que representam a anotação sugerida pelo BioAgents-Prot, e $B = \{b_1, b_2, \ldots, b_m\}$ o conjunto de palavras representando a anotação manual, a similaridade entre os dois conjutnos de anotação de proteínas é estimada com o *coeficiente de sobreposição* baseado no *índice de Jaccard*, como mostrado na Equação 5.1:

$$O(A, B) = \frac{|A \cap B|}{min(|A|, |B|)} \tag{5.1}$$

Nesta equação, se A é um subconjunto de B, ou o contrário, então o coeficiente de sobreposição é igual a um, sugerindo que ambos os conjuntos compartilham a mesma anotação. Além disso, as anotações foram classificadas como iguais quando $O(A, B) \geq 66\%$, significando que uma razão de 2/3 entre as anotações indicam que são iguais.

O *Registro de Conceitos Suplementares*, parte do thesaurus *Medical Subject Headings* (MeSH) [7], foi utilizado para refinar o coeficiente de sobreposição. Este registro corresponde à categoria de termos *Química e Drogas* do MeSH, composta principalmente de conceitos de proteínas e reações químicas. Cada conceito apresenta uma lista de nomes alternativos de proteínas ou reações químicas, úteis para auxiliar a comparação entre duas anotações.

Para obter uma lista de nomes alternativos, comuns entre duas anotações, o conjunto $A \cap B$ é utilizado como consulta no *Registro de Conceitos Suplementares*. A lista de nomes alternativos obtida com a consulta é utilizada na tentativa de "casar" elementos que estejam excluídos do conjunto $A \cap B$. A partir do casamento entre anotações alternativas e os conjuntos A e B, dois coeficientes de sobreposição são obtidos:

- C_A: A é sobreposto tanto por palavras de B quanto por palavras de nomes de proteína alternativos à de $A \cap B$; e
- C_B: B é sobreposto tanto por palavras de A quanto por palavras de nomes de proteína alternativos à de $A \cap B$.

O valor máximo entre os coeficientes C_A e C_B é utilizado como medida para estimar a similaridade entre as duas anotações.

Exemplo

Sejam *">SCRT_00014 | Saccharomyces cerevisiae RM11-1a* ***ubiquinol cytochrome C oxidoreductase subunit 7*** *(384 nt)"*, a anotação manual, e *"sp|P00128| QCR7_YEAST* ***Cytochrome b-c1 complex subunit 7*** *OS=Saccharomyces cerevisiae (strain ATCC 204508 / S288c) GN=QCR7 PE=1 SV=2"*, a recomendação de anotação do BioAgents-Prot. A parte textual que contém a informação de anotação é destacada e filtrada, formando, em seguida, dois conjuntos:

- A = {ubiquinol, **cytochrome**, c, oxidoreductase, **subunit**}; e
- B = {**cytochrome**, b, c1, complex, **subunit**}.

A interseção $A \cap B$ = {subunit, cytochrome} é formada pelas palavras que são comuns entre as anotações A e B.

Como os coeficientes de sobreposição $\frac{|A \cap B|}{|A|}$ e $\frac{|A \cap B|}{|B|}$ são $< \frac{2}{3}$, o conjunto $A \cap B$ é utilizado como consulta no *Registro de Conceitos Suplementares* (RCS), com o intuito de buscar por anotações que são alternativas aos conjuntos A e B.

Ao ser realizada a consulta a partir dos elementos da interseção $A \cap B$, a seguinte lista será obtida:

⋮
- PsbE protein, Arabidopsis
  - **cytochrome** b-559 alpha **subunit**, Arabidopsis
- QCR6 protein, S cerevisiae
  - **cytochrome** c reductase **subunit** 6, S cerevisiae
  - **subunit** 6, **cytochrome** c reductase, S cerevisiae
- **Qcr7 protein, S cerevisiae**
  - **ubiquinol-cytochrome c oxidoreductase subunit 7 protein, S cerevisiae**

Nesta lista, observa-se que as palavras *"cytochrome"* e *"subunit"* casaram com anotações alternativas aos elementos de $A \cap B$.

Cada anotação identificada no RCS é uma alternativa para os elementos de $A \cap B$. Essa anotação identificada possui um registro pai, que contém uma lista de outras anotações alternativas. Como exemplo, a anotação identificada como *"****cytochrome*** *c reductase* ***subunit*** *6, S cerevisiae"* possui um registro pai, com anotação *"QCR6 protein, S cerevisiae"*, que inclui em sua lista uma segunda anotação alternativa, identificada como **subunit** *6,* **cytochrome** *c reductase, S cerevisiae.*

Cada registro pai e sua lista de nomes alternativos são filtrados e representados em conjuntos para posterior cálculo de sobreposição em relação aos conjuntos A e B. Nesta consulta, o valor máximo de sobreposição obtido foi de 100%, através do registro *"Qcr7*

protein, S cerevisiae". Esse registro e o seu nome alternativo são representados nos conjuntos de anotação C e D, como segue:

- C = {qcr7}.
- D = {ubiquinol, cytochrome, c, oxidoreductase, subunit}.

Os coeficientes resultantes de sobreposição, com auxílio desses dois conjuntos são:

$$\frac{|A \cap (B \cup C \cup D)|}{|A|} = \frac{5}{5} = 100\%, \; ou \; \frac{|B \cap (A \cup C \cup D)|}{|B|} = \frac{2}{5} = 40\%$$

Uma vez que os conjuntos B, C e D sugerem anotações alternativas ao conjunto A, o coeficiente de sobreposição em relação a A de 100% sugere que as anotações de A e B são as mesmas.

5.4 Critérios de performance

Para avaliar a performance do BioAgents-Prot, em relação ao padrão ouro, foram utilizadas as seguintes métricas: sensibilidade (*recall*), especificidade, F1-score e o coeficiente de correlação de Matthews (MCC) [51].

- Sensibilidade: expressa a capacidade de anotar corretamente transcritos como proteínas;
- Especificidade: mede a capacidade de não anotar como proteínas sequências que não são;
- F1-score: mede a acurácia do sistema; e
- Coeficiente de Correlação de Matthews (MCC): mede o coeficiente de correlação entre as classificações preditas e observadas.

As medidas F1-score e MCC foram utilizadas como medida de desempenho do BioAgents-Prot, pois os conjuntos de negativos e positivos não estão balanceados. Neste caso (quando duas classes possuem tamanhos muito diferentes), outros métodos, como a acurácia, não são adequados.

F1-score corresponde ao *Coeficiente de Dice* da teoria dos conjuntos, sendo definido pela média harmônica normalizada para a classe de positivos, onde VP são os *Verdadeiros*

Positivos, FN são os *Falsos Negativos*, TN são os *Verdadeiros Negativos* e FP são os *Falsos Positivos*, conforme apresentado na Equação 5.2.

$$F1 = \frac{2 \times VP}{(2 \times VP) + FP + FN} \tag{5.2}$$

MCC corresponde ao coeficiente de correlação de Pearson aplicado à matriz de contingência [64], conforme apresentado na Equação 5.3.

$$MCC = \frac{(VP \times VN) - (FP \times FN)}{\sqrt{(VP + FN)(VN + FP)(VP + FP)(VN + FN)}} \tag{5.3}$$

O coeficiente MCC varia entre -1 e 1, onde -1 indica uma correlação negativa perfeita, 0 indica uma distribuição randômica e 1 indica uma correlação positiva perfeita [51].

5.5 Resultados

Dentre os resultados do BioAgents-Prot, foram obtidas anotações com as seguintes qualidades:

- confiável: anotação do GRH confirmada (textualmente) com domínios conservados com $escore \geq 70\%$;
- moderada: anotação do GRH confirmada (textualmente) com domínios conservados com $50\% \leq score \leq 70\%$; e
- inferida por similaridade: anotação do GRH com $identidade \geq 70\%$ e $positivos \geq 80\%$, mas sem confirmação com domínios conservados.

Um total de 5.694 transcritos foram analisados pelo BioAgents-Prot (Figura 5.2). Destes: (i) 5.579 transcritos foram anotados como proteína, sendo 3.725 confiáveis, confirmados com domínios conservados, 880 apresentaram uma fraca confirmação com domínios conservados, e 974 foram inferidos apenas por similaridade de sequência; e (ii) 115 transcritos não foram anotados. Por outro lado, de 413 sequências de ncRNAs conhecidas, 28 foram anotadas como proteína pelo BioAgents-Prot e 385 não foram.

As anotações do BioAgents-Prot foram comparadas com as anotações manuais dos transcritos da *S. cerevisiae*, utilizando o cálculo de similaridade funcional entre anotações, explicado na Seção 5.3. Como o padrão ouro apresenta $36,63\%$ atribuições de funções putativa e proteínas hipotéticas, não é fácil utilizar esta porção como referência para verificar se as sugestões do BioAgents-Prot sugerem a mesma anotação, porque estas anotações indicam que eles codificam uma proteína, mas continuam com uma função desconhecida. Neste caso, se o BioAgents-Prot sugere uma anotação, e se a anotação

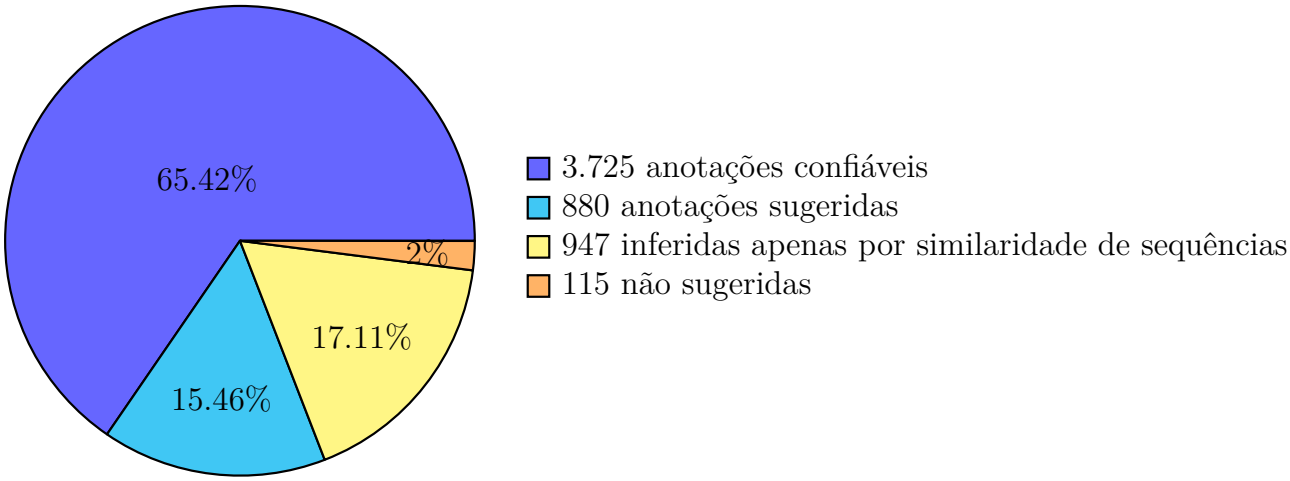


Figura 5.2: Anotação do BioAgents-Prot dos transcritos da *S. cerevisiae*, onde 65, 42% corresponde a 3.725 anotações confiáveis, 15, 46% corresponde a 880 anotações que apresentaram uma fraca confirmação com domínios conservados, 17, 11% correspondem a 947 anotações inferidas apenas por similaridade de sequência, e 2% corresponde a 115 transcritos com nenhuma sugestão.

manual é *"unknown function"* ou por exemplo *"hypothetical protein similar to Rtn1p"*, consideramos a anotação do BioAgents-Prot como correta (correspondendo a um VP).

Exemplos de funções putativas, anotadas manualmente na *S. cerevisiae*:

- *hypothetical protein similar to Rtn1p;*
- *hypothetical protein similar to glucosidase II beta subunit;*
- *hypothetical protein similar to member of the Sir2 family of NAD(+)-dependent protein deacetylases;*
- *hypothetical protein similar to Pontin52;*
- *Uncharacterized protein YDR179W-A;*
- *hypothetical protein similar to Nap1-binding protein;*
- *hypothetical protein similar to interacts with PP2C;*
- *hypothetical protein similar to component of the SPS plasma membrane amino acid sensor system;*
- *hypothetical protein similar to dihydrolipoyl transsuccinylase, mitochondrial;*
- *hypothetical protein similar to F45H11.2.*

Acreditamos que essas anotações foram atribuídas da mesma maneira que o BioAgents-Prot, por similaridade de sequências, pois sugerimos uma anotação com função de outra

proteína. Deve-se notar que não estamos certos se a anotação inferida realiza a mesma função, sem fortes evidências que confirmem a anotação, como explicado na Seção 2.2.1.

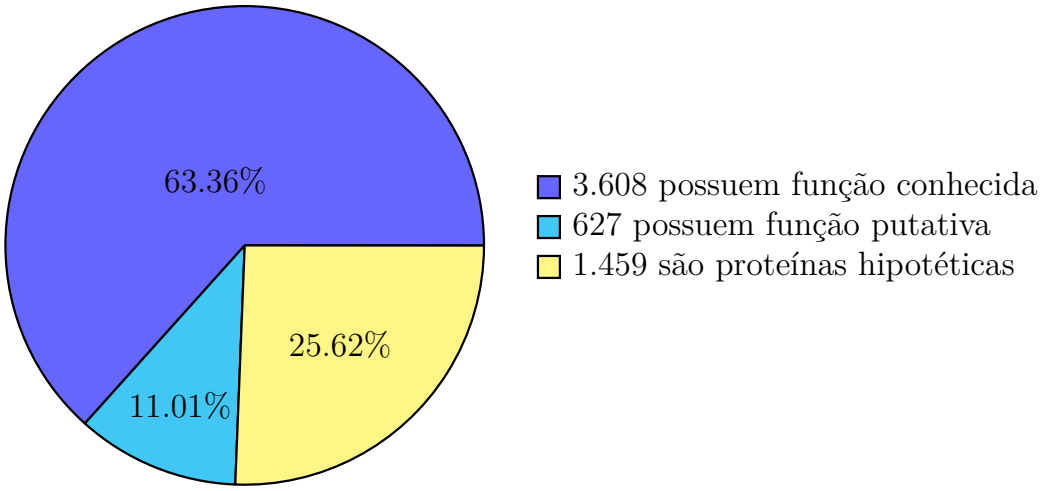


Figura 5.3: Anotação manual dos transcritos da *S. cerevisiae*, onde 63, 36% de funções conhecidas correspondem à 3.608 transcritos, 11, 01% de funções putativas corrrespondem à 627 transcritos, e 25, 62% de proteínas hipotéticas correspondem à 1.459 transcritos.

Utilizando o cáculo de similaridade funcional e o critério descrito para computar VP, de 5.579 anotações do BioAgents-Prot, 5.050 foram considerados como VP e 529 apresentaram imcompatibilidade em relação à anotação manual. Estas 529 anotações requerem uma comparação manual, devido às anotações que não sobrepõem as anotações manuais, mas semanticamente podem corresponder à mesma função, por exemplo, a anotação manual *"mitochondrial distribution and morphology protein 39"* é um nome alternativo à anotação do BioAgents-Prot *"Golgi to ER traffic protein 1"*. Nossa comparação manual conclui que 492 destas anotações possuem a mesma função e 37 são diferentes.

Na matriz de contigência (Tabela 5.1), o conjunto de positivos (P) é formado por todos os 5.694 transcritos, enquanto o conjunto de negativos (N) inclui todas as 413 sequências de ncRNAs. Para o conjunto P, 5.050 mais 492 comparações manuais identificadas como mesma função totalizam 5.542 VP, e 37 anotações diferentes mais 115 anotações não atribuídas totalizam 152 FN. Para o conjunto N, 28 sequências de ncRNAs atribuídas como proteína são FP e as outras 385 sequências sem sugestão de anotação são VN.

Com base na matriz de contingência, o BioAgents-Prot apresentou as seguintes medidas de performance: 95, 84% de *sensibilidade*, 93, 22% de *especificidade*, 98, 40% de *F1-score* e 0, 80 de *MCC*. Estas medidas mostram que BioAgents-Prot possui alta performance para anotar proteínas adequadamente. As medidas F1-score e MCC avaliam o quanto o BioAgents-Prot é capaz de atribuir funções às proteínas adequadamente, evitando uma classificação errada, tendo apresentado um *F1-score* muito próximo de 100%

Tabela 5.1: Matriz de contingência produzida com transcritos e ncRNAs da *S. cerevisiae*.

Análise	P 5.694 transcritos	N 413 ncRNAs
Sugestões positivas do BioAgents-Prot	5.542 (VP)	28 (FP)
Sugestões negativas do BioAgents-Prot	152 (FN)	385 (VN)

e uma correlação muito forte, como indicado pela pontuação do MCC ser muito próxima de uma correlação positiva perfeita ($MCC = 1$).

Capítulo 6

Conclusões e trabalhos futuros

Neste trabalho, criamos o BioAgents-Prot, uma ferramenta multiagente baseada em conhecimento para anotação de proteínas em projetos transcritoma, buscando obter anotações confiáveis, a partir de regras de anotação bem definidas. Em particular, definimos uma nova arquitetura, que foi implementada utilizando o *framework* JADE e o motor de inferência Drools. Uma interface web amigável foi criada, que permite a visualização das recomendações de anotação. Para validar o protótipo, realizamos um experimento com o fungo *Saccharomyces cerevisiae* para medir o desempenho do sistema. Quando comparados com a anotação manual da *S. cerevisiae*, obtivemos 95, 84% de *sensibilidade*, 93, 22% de *especificidade*, 98, 40% de *F1-score* e $0,80$ de *MCC*. Pode-se perceber também, que BioAgents-Prot forneceu $65,42\%$ de anotações consideradas confiáveis, enquanto a anotação manual forneceu $63,36\%$ de funções conhecidas.

6.1 Contribuições

A maior contribuição deste trabalho é utilizar agentes de forma integrada, para recomendar anotações de forma mais confiável. Em particular, modificamos a arquitetura proposta inicialmente por Lima [48], com foco em anotação de proteínas em transcritos, incluindo características de domínios conservados e estudo de conservação em organismos relacionados filogeneticamente. Além disso, propusemos um cálculo de comparação de anotações (da recomendação do BioAgents-Prot e da manual) baseado em termos retirados do *Registro de Conceitos Suplementares*, que integra o thesaurus MeSH. Por fim, foi desenvolvido um sistema, disponível publicamente, com uma interface amigável.

Um resumo (Anexo B) foi apresentado no congresso *X-meeting*, promovido pela AB3C (Associação Brasileira de Bioinformática e Biologia Computacional).

6.2 Trabalhos futuros

Sugerimos as seguintes extensões:

- aprimoramento do raciocínio do sistema:
  - refinar as regras existentes;
  - incluir novos métodos e estratégias de anotação de proteínas;
  - incluir um conjunto de agentes mineradores, que poderiam combinar diferentes resultados, para extrair informações úteis que refinem a anotação;
  - incluir mecanismos de descoberta de conhecimento e aprendizado, que permitam ampliar a autonomia dos agentes.
- aprimoramento do desempenho do sistema:
  - melhorar a forma de armazenamento de dados;
  - realizar uma implementação distribuída, que permita diminuir o tempo de execução do BioAgents-Prot.

Referências

[1] Apache Jena. Disponível em: http://jena.apache.org/. Acessado em: Dezembro de 2014. 24, 28

[2] BLAST databases. Disponível em: http://www.ncbi.nlm.nih.gov/blast/blast_databases.shtml. Acessado em: Dezembro de 2014. 15, 35

[3] ClustalW Tutorial. *The Southwest Biotechnology and Informatics Center - SWBIC*. Disponível em: http://outreach.gtldna.com/origin/proc_man/Clustal/Clustal_tutorial.html. 41

[4] Espécies de Candida - Broad Institute. Disponível em: http://www.broadinstitute.org/annotation/genome/candida_albicans/MultiDownloads.html. Acessado em: Dezembro de 2014. 47

[5] Espécies de Schizosaccharomyces - Broad Institute. Disponível em: http://www.broadinstitute.org/annotation/genome/schizosaccharomyces_group/MultiDownloads.html. Acessado em: Dezembro de 2014. 47

[6] Figura do fungo Saccharomyces cerevisiae. Disponível em: http://microbewiki.kenyon.edu/index.php/File:Saccromyces.jpg. Acessado em: Dezembro de 2014. xi, 47

[7] Medical Subject Headings - MeSH. Disponível em: http://www.nlm.nih.gov/mesh/. Acessado em: Dezembro de 2014. 48

[8] Saccharomyces cerevisiae RM11-1a Database - Broad Institute. Disponível em: http://www.broadinstitute.org. Acessado em: Dezembro de 2014. 47

[9] Saccharomyces cerevisiae strain S288C - broad Institute. Disponível em: http://www.yeastgenome.org. Acessado em: Dezembro de 2014. 47

[10] B. Alberts, D. Bray, K. Hopkin, et al. *Fundamentos da Biologia Celular*. ARTMED, 3rd edition, 2011. x, 9

[11] S. F. Altschul, W. Gish, W. Miller, et al. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, October 1990. 14, 41

[12] A. Andreeva, D. Howorth, S. E. Brenner, et al. SCOP database in 2004: refinements integrate structure and sequence family data. *Nucleic Acids Research*, 32(suppl 1):D226–D229, 2004. 17

[13] M. Ashburner, C. A. Ball, J. A. Blake, et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature genetics*, 25(1):25–29, May 2000. 12

[14] D. Batory. The LEAPS Algorithm. Technical report, Austin, TX, USA, 1994. 28

[15] F. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-Agent Systems with JADE*. Wiley Series in Agent Technology, 2007. 4, 30

[16] B. Bergeron. *Bioinformatics Computing*. Prentice Hall PTR, November 19, 2002. x, 9

[17] H. M. Berman, J. Westbrook, Z. Feng, et al. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000. 17

[18] M. Bhagwat, L. Young, and R. R. Robison. Using BLAT to find sequence similarity in closely related genomes. *Current protocols in bioinformatics / editoral board, Andreas D. Baxevanis ... [et al.]*, Chapter 10:Unit10.8, March 2012. 14

[19] B. Boeckmann, A. Bairoch, R. Apweiler, et al. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic acids research*, 31(1):365–370, January 2003. 15, 35

[20] P. Bork, T. Dandekar, Y. Diaz-Lazcoz, et al. Predicting function: from genes to genomes and back. *Journal of Molecular Biology*, 283(4):707 – 725, 1998. x, 11, 12

[21] P. Clote. *Computational Molecular Biology - An introduction*. John Wiley & Sons Ltd, 2000. 6

[22] J. Collis, D. Ndumu, and C. van Buskrik. The ZEUS technical manual. *Intelligent Systems Research Group, BT Labs, British Telecommunications*, 1999. 31

[23] C. S. da F. Filho. JEOPS – Integração entre Objetos e Regras de Produção em Java. Master's thesis, Centro de Informática, Universidade Federal de Pernambuco (UFPE), 2000. 28

[24] C. Darwin. *The Origin of Species*. p. F. Collier & Son, 1909. 13, 18

[25] J. Daugelaite, A. O' Driscoll, and R. D. Sleator. An Overview of Multiple Sequence Alignments and Cloud Computing in Bioinformatics. *ISRN Biomathematics*, 2013(2):14, 2013. 18

[26] K. Decker, X. Zheng, and C. Schmidt. A multi-agent system for automated genomic annotation. In *AGENTS'01: Proceedings of the 5th International Conference on Autonomous Agents*, pages 433–440, New York, NY, USA, 2001. ACM. 20

[27] L. Ding, A. Sabo, N. Berkowicz, et al. EAnnot: A genome annotation tool using experimental evidence. *Genome Research*, 14(12):2503–2509, December 2004. 19

[28] R. B. Doorenbos. Production Matching for Large Learning Systems. Technical report, 1995. 28

[29] S. R. Eddy. Profile hidden markov models. *Oxford Journals, Bioinformatics*, 1998. 15

[30] S. R. Eddy and T. J. Wheele. *HMMER User's Guide*. Janelia Farm Research Campus, May 2013. 15

[31] T. Etzold and P. Argos. SRS–an indexing and retrieval tool for flat file data libraries. *Computer applications in the biosciences : CABIOS*, 9(1):49–57, 1993. 37

[32] FIPA. Foundation for Intelligent Physical Agents, 2014. [Online; acessado em Fevereiro de 2014]. 29

[33] K. Forslund and E. L. L. Sonnhammer. Predicting protein function from domain content. *Bioinformatics*, 24(15):1681–1687, 2008. 19

[34] J. A. Gerlt and P. C. Babbitt. Can sequence determine function? *Genome biology*, 1(5), 2000. 13

[35] J. C. Giarratano and G. D. Riley. *Expert Systems: Principles and Programming*. Course Technology, 3rd edition, 1998. x, xii, 25, 27

[36] O. Gutknecht and J. Ferber. The MadKit Agent Platform Architecture. In *In Agents Workshop on Infrastructure for Multi-Agent Systems*, pages 48–55, 2000. 30

[37] E. F. Hill. *Jess in Action: Java Rule-Based Systems*. Manning Publications Co., Greenwich, CT, USA, 2003. 27, 28

[38] M. Iacono, L. Villa, D. Fortini, et al. Whole-Genome Pyrosequencing of an Epidemic Multidrug-Resistant Acinetobacter baumannii Strain Belonging to the European Clone II Group. *Antimicrobial Agents and Chemotherapy*, 52(7), 2008. doi:10.1128/AAC.01643-07. 19

[39] M. Janitz. *Next-Generation Genome Sequencing: Towards Personalized Medicine*. WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim, 2008. 1, 11

[40] W. J. Kent. BLAT–the BLAST-like alignment tool. *Genome research*, 12(4):656–664, April 2002. 14

[41] T. Koestler, A. von Haeseler, and I. Ebersberger. FACT: functional annotation transfer between proteins with similar feature architectures. *BMC Bioinformatics*, 11(1):417+, 2010. 19

[42] R. Kowalski. Robert Kowalski: A Short Story of My Life and Work. `http://www.doc.ic.ac.uk/~rak/history.html`. 28

[43] R. Kowalski. Predicate logic as programming language. *IFIP congress*, 74:569–544, 1974. 25

[44] R. Kowalski. *Computational logic and human thinking : how to be artificially intelligent*. Cambridge University Press, 2011. 25, 26

[45] R. Kowalski and D. Kuehner. Linear Resolution with Selection Function. *Artificial Intelligence*, 2(3-4):227–260, December 1971. 28

[46] E. S. Lander, L. M. Linton, B. Birren, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001. 1

[47] D. Leja. Artist from National Human Genome Research Institute (NHGRI) http://www.genome.gov/12514471. x, 8

[48] R. S. Lima. Sistema Multiagente para Anotação Manual em Projetos de Sequenciamento de Genomas. Master's thesis, Departamento de Ciência da Computação, Universidade de Brasília (UnB), 2007. 3, 55

[49] R. S. Lima, C. G. Ralha, H. W. Schneider, et al. BioAgents: Um Sistema Multiagente para Anotação Manual em Projetos de Seqüênciamento de Genomas. In *Anais do VI Brazilian Meeting on Artificial Intelligence - ENIA*, pages p. 1302–1310, Rio de Janeiro, Brasil, 2007. 3

[50] N. F. Martins, M. E. M. T. Walter, G. P. Telles, and M. M. Brigido, editors. *III Brazilian Workshop on Bioinformatics, October 20-22, 2004, Brasília, Distrito Federal, Brazil*, 2004. 60, 61

[51] B. W. Matthews. Comparison of the predicted and observed secondary structure of {T4} phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442 – 451, 1975. 50, 51

[52] G. Mendel. Experiments in Plant Hybridization. *Proceedings of the Brünn Natural History Society*, IV, 1865. 13

[53] D. Merritt. Using Prolog's Inference Engine. In *Building Expert Systems in Prolog*, Springer Compass International, pages 15–31. Springer New York, 1989. 28

[54] J. R. Miller, S. Koren, and G. Sutton. Assembly Algorithms for Next-Generation Sequencing Data. *Genomics*, 2010. Author manuscript; available in PMC 2011 June 1. 2

[55] M. J. Moore, A. Dhingra, P. S. Soltis, et al. Rapid and accurate pyrosequencing of angiosperm plastid genomes. *BMC Plant Biology*, 6(17), 2006. doi:10.1186/1471-2229-6-17. 19

[56] D. W. Mount. Using the Basic Local Alignment Search Tool (BLAST). *Cold Spring Harbor Protocols*, 2007(7):pdb.top17, 2007. 14

[57] L. V. Nascimento and A. L. C. Bazzan. An Agent-Based System for Re-annotation of Genomes. In Martins et al. [50], pages 41–48. 20

[58] Y. Ofran, M. Punta, R. Schneider, et al. Beyond annotation transfer by homology: novel protein-function prediction methods to assist drug discovery. *Drug Discovery Today*, 10(21):1475 – 1482, 2005. 13

[59] C. A. Orengo, J. E. Bray, D. W. A. Buchan, et al. The CATH protein family database: A resource for structural and functional annotation of genomes. *PROTEOMICS*, 2(1):11–21, 2002. 17

[60] A. Orro and L. Milanesi. An agent approach for protein function analysis in a grid infrastructure. *Stud Health Technol Inform*, 126:314–321, 2007. 19

[61] G. Pandey, V. Kumar, and M. Steinbach. Computational Approaches for Protein Function Prediction: A Survey. Technical report, Department of Computer Science and Engineering, University of Minnesota, Twin Cities, 2006. 2, 11, 13, 16, 18

[62] W. R. Pearson. An introduction to sequence similarity ("homology") searching. *Current protocols in bioinformatics / editoral board, Andreas D. Baxevanis ... [et al.]*, Chapter 3:Unit3.1, June 2013. 13

[63] A. Pokahr, L. Braubach, and W. Lamersdorf. Jadex: Implementing a BDI-Infrastructure for JADE Agents. *EXP*, 3(3):76–85, September 2003. 30

[64] D. M. W. Powers. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *International Journal of Machine Learning Technology*, 2(1):37–63, 2011. 51

[65] M. Proctor. Drools Business Rule Management System (BRMS). `http://www.jboss.org/drools/`. 4, 27, 36

[66] K. D. Pruitt, T. Tatusova, and D. R. Maglott. NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic Acids Research*, 33(suppl 1):D501–D504, 2005. 15

[67] M. Punta, P. C. Coggill, R. Y. Eberhardt, et al. The Pfam protein families database. *Nucleic Acids Research*, 40(D1):D290–D301, January 2012. 16

[68] P. Radivojac, W. T. Clark, T. R. Oron, et al. A large-scale evaluation of computational protein function prediction. *Nature Methods*, 10(3):221–7, 2013. ID: unige:33780. 2, 3, 13, 37

[69] C. G. Ralha, H. W. Schneider, L. Fonseca, et al. Using BioAgents for Supporting Manual Annotation on Genome Sequencing Projects. In *Lecture Notes on Bioinformatics (LNBI)*, pages v. 5167. p. 127–139, São Paulo, Brasil, 2008. Springer. Apresentado em: Brazilian Syposium on Bioinformatics (BSB 2008). 3

[70] C. G. Ralha, H. W. Schneider, M. E. M. T. Walter, et al. Reinforcement Learning Method for BioAgents. In *Proceedings of the Brazilian Symposium on Artificial Neural Network-SBRN 2010*, pages p. 109–114, 2010. 3

[71] S. Russell and P. Norvig. *Artificial intelligence: a modern approach (3rd edition)*. Prentice Hall Series, 2009. x, 21, 22, 23, 24, 25, 26

[72] C. T. Santos and A. L. C. Bazzan. Using the A3C System for Annotation of Keywords - A Case Study. In Martins et al. [50], pages 175–178. 19

[73] H. W. Schneider. Método de Aprendizagem por Reforço no Sistema Bioagents. Master's thesis, Departamento de Ciência da Computação, Universidade de Brasília (UnB), 2010. 3

[74] F. Servant, C. Bru, S. Carrère, et al. ProDom: Automated clustering of homologous domains. *Briefings in Bioinformatics*, 3(3):246–251, 2002. 14, 15, 35, 36, 37

[75] J. C. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology*. Brooks/Cole Publishing Company, Pacific Grove, CA, 1997. x, 7

[76] J. Shrager. The fiction of function. *Bioinformatics*, 19(15):1934–1936, 2003. 11

[77] F. Sievers, A. Wilm, D. Dineen, et al. Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. *Molecular Systems Biology*, 7(1):n/a–n/a, 2011. 18

[78] G. Solda, I. V. Makunin, O. U. Sezerman, et al. An Ariadne's thread to the identification and annotation of noncoding RNAs in eukaryotes. *Brief Bioinform*, 10(5):475–489, September 2009. 19

[79] D. S. Souza. BioAgents: uma ferramenta multiagente para anotação de sequências biológicas. Monografia de bacharelado, Departamento de Ciência da Computação, Centro Universitário IESB, 2012. 3

[80] W. J. Tolone, D. Wilson, A. Raja, et al. Applying Cougaar to Integrated Critical Infrastructure Modeling and Simulation. In *Proceedings of the 1st Open Cougaar Conference*, pages 3–10, New York City, USA, 2004. 30

[81] A. Tramontano. *The Ten Most Wanted Solutions in Protein Bioinformatics*. Chapman & Hall/CRC mathematical biology and medicine series. Chapman & Hall/CRC, 1st edition, 2005. x, 2, 13

[82] J. Craig Venter, M. D. Adams, E. W. Myers, , et al. The Sequence of the Human Genome. *Science*, 291(5507):1304–1351, 2001. 1

[83] N. Vlassis. *A concise introduction to multiagent systems and distributed artificial intelligence*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers, 1st edition, 2007. 21

[84] Z. Wang, M. Gerstein, and M. Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, January 2009. 19

[85] J. D. Watson and F. H. Crick. Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature*, 1953. 7, 8

[86] J. D. Watson, A. Gann, T. A. Baker, et al. *Molecular Biology of the Gene*. Pearson Education, Inc, 7 edition, 2014. 10

[87] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Modern Approach to Artificial Intelligence*. The MIT Press, 3rd edition, 1999. 21, 22

[88] M. Wooldridge. *An Introduction to Multiagent Systems*. John Wiley & Sons Ltd, 2 edition, 2002. 21, 22, 30

[89] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10:115–152, 1995. 21

[90] D. Xavier, B. Crespo, R. Fuentes-Fernández, et al. MASSA: Multi-Agent System to Support Functional Annotation. In Yves Demazeau, Franco Zambonelli, JuanM. Corchado, and Javier Bajo, editors, *Advances in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection*, volume 8473 of *Lecture Notes in Computer Science*, pages 291–302. Springer International Publishing, 2014. 19

Anexo A

Criação do banco ProDom

Os arquivos FASTA necessários para criação do ProDom são: (i) arquivo de regiões conservadas, provenientes do alinhamento múltiplo de cada família; e (ii) arquivo do consenso das regiões conservadas de cada família. Por engenharia reversa (realizada para esse procedimento), esses arquivos foram reconstruídos a partir do banco de indexação no formato SRS, fornecido pelo *site* do ProDom.

O *script* a seguir foi desenvolvido na linguagem Perl para criação desses dois arquivos, descritos acima.

```
#!/usr/bin/perl -w

#~ input
open (INP, "<./current_release/prodom.srs");
#~ output in FASTA format
open (MUL, ">./current_release/prodom.mul");
open (CON, ">./current_release/prodom.cons");

while(not eof(INP)) {
  my $ac;
  while(<INP>) {
    if ($_ =~ m/^AC\s+(\S+)\s*/) { $ac = $1; last; }
  }

  my $desc;
  while(<INP>) {
    if ($_ =~ m/^KW.+\/\/\s+(.+)\s*$/) { $desc = $1; last; }
  }

  my $count = 0;
  my @array;
  my $co;
  while(<INP>) {
```

```
    if ($_ =~ m/^CO\s+(\S+)\s*/) {
      $co = $1;
      $co =~ s/-|\.//g;
      #~ consensus of each family formed by MSA of several organisms:
      #~ print the header and next the sequence
      print CON ">CONSENSUS#$ac | ".length($co)." | pd_$ac; | ($count) $desc\n";
      print CON join("\n", unpack ("(A59)*", $co))."\n";
      last;
    } elsif($_ =~ m/^AL/) {
      push (@array, $_);
      $count++;
    }
  }

  #~ For each subsequence of the same family
  foreach (@array) {
    $_ =~ m/^AL\s+(\S+)\|(\S+)\s+(\S+)\s+(\S+)\s+\S+\s+(\S+)\s*/;
    #~ print the header
    print MUL ">$2#$ac#$3#$4 | ".($4 - $3 + 1)." | pd_$ac;sp_$2_$1; | ($count) $desc\n";
    my $seq = $5;
    $seq =~ s/-|\.//g;
    #~ print the subsequence
    print MUL join("\n", unpack ("(A59)*", $seq))."\n";
  }
  undef @array;
}

#~ close files
close (INP);
close (MUL);
close (CON);

exit 0;
```

A partir destes dois arquivos, foram construídos dois bancos indexados para uso com a ferramenta BLAST, utilizados também no *script* BlastProDom. A construção dos bancos foi realizada com a ferramenta *makeblastdb*, utilizando os seguintes parâmetros:

```
makeblastdb -dbtype prot -out prodom.mul -title "ProDom Release  2010.1"
-logfile prodom.mul.log -parse_seqids -hash_index -in prodom.mul
```

```
makeblastdb -dbtype prot -out prodom.cons -title "ProDom Release  2010.1"
-logfile prodom.cons.log -parse_seqids -hash_index -in prodom.cons
```

Anexo B

BioAgents-Prot: a multiagent tool to annotate proteins

DS Souza[1], RC Togawa[3], NF Martins[3], P Grynberg[3],
TR Alencar[2], CG Ralha[1], MEMT Walter[1]

[1]Departament of Computer Science, University of Brasília (UnB)
[2]Department of Cellular Biology, University of Brasília (UnB)
[3]Laboratory of Bioinformatics, Genetic Resources and Biotechnology - CENARGEN/EMBRAPA

e-mails: {dssouzadan, tainaraiol}@gmail.com, {mia, ghedini}@cic.unb.br,
{roberto.togawa, natalia.martins, priscila.grynberg}@embrapa.br

Abstract

Background: An important task when analyzing genomic sequences is the identification of their functions and biological characteristics. This task is basically done by homology (using algorithms of sequence comparison) and biologists' knowledge, such that functions of the sequences of an organism of interest can be predicted from similar sequences with already determined functions.
Objectives: This work has the objective of proposing BioAgents-Prot, a multiagent system (MAS) capable of annotating proteins, based on results of comparison sequence algorithms together with production rules that formalize biologists' reasoning.
Methods: The BioAgents-Prot architecture is divided in three layers. The interface layer models the system interface between the user and BioAgents-Prot. The collaborative layer is the MAS core, composed of manager and analyst agents, where each manager agent simulates biologists' reasoning, using an associated knowledge source (KS) with a rule set to deal with results parsed by their analysts. Agents are grouped according to the method of annotation (e.g., inference by homology or conservation among phylogenetic related organisms). In particular, an annotation manager was proposed, which creates manager agents according to the following. First, annotation based on homology (Blast with databases nr-fungi and SwissProt) is tried, and it can be confirmed with conserved domains (Blast with database ProDom), or not. In this last case, annotation may be obtained (or confirmed) by investigating conservation among related organisms (ClustalW). Finally, the physical layer is composed of public available genomic databases.
Results: We performed a case study with the *Saccharomyces cerevisiae* fungus: 5.694 protein

sequences were downloaded from *http://www.broadinstitute.org* and 413 non-coding RNAs were downloaded from *http://www.yeastgenome.org*. We compared the Broad and the Yeast Genome annotations of each sequence with the annotation suggested by BioAgents-Prot, having obtained 74.15% of sensitivity (*True Positives/True Positives + False Negatives*), $96,13\%$ of specificity (*True Negatives/True Negatives + False Positives*) and $75,63\%$ of accuracy (*True Positives + True Negatives/Positives + Negatives*). These results show that BioAgents-Prot annotation is reliable, in the sense that it can not identify about $25,85\%$ of proteins, but it does not recommend false positives ($96,13\%$). These analyses were developed considering that the annotation in both annotation files of *S. cerevisiae* (Broad and Yeast Genome) are correct, but we have to investigate them more deeply.

Conclusion: BioAgents-Prot is a MAS to suggest annotation for transcript sequences. It formalizes biological reasoning with a knowledge base of production rules. As future work, we intend to improve sensibility (and then accuracy) of our system, maybe including more manager and analyst agents for protein *de novo* prediction. Besides, *argumentation theory* can give to the agents the ability to argue and influence the overall annotation.

Keywords: transcript annotation, multiagent environment, biologists' reasoning, production rules.

Supported by: CNPq