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

ncRNA-Agents: Anotação de RNAs não-codificadores Baseada em Sistema Multiagente

Wosley da Costa Arruda

Tese apresentada como requisito parcial
para conclusão do Doutorado em Informática

Orientadora
Prof.ª Dr.ª Maria Emilia M. T. Walter

Brasília
2015

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

Coordenadora: Prof.ª Dr.ª Alba Cristina M. A. de Melo

Banca examinadora composta por:

Prof.ª Dr.ª Maria Emilia M. T. Walter (Orientadora) — CIC/UnB
Prof. Dr. Nalvo F Almeida Jr — Faculdade de Computação/UFMS
Prof. Dr. Roberto Coiti Togawa — EMBRAPA/DF
Prof. Dr. Marcelo de Macedo Brígido — IB/UnB
Prof.ª Dr.ª Célia Ghedini Ralha — CIC/UnB

CIP — Catalogação Internacional na Publicação

da Costa Arruda, Wosley.

ncRNA-Agents: Anotação de RNAs não-codificadores Baseada em Sistema Multiagente / Wosley da Costa Arruda. Brasília : UnB, 2015.
139 p. : il. ; 29,5 cm.

Tese (Doutorado) — Universidade de Brasília, Brasília, 2015.

1. RNAs não-codificadores, 2. Anotação de RNAs não-codificadores,
3. Sistema Multiagente, 4. Bioinformática, 5. Inteligência Artificial

CDU 004.8

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

WOSLEY DA COSTA ARRUDA

NCRNA-AGENTS: ANOTAÇÃO DE RNAS NÃO-CODIFICADORES BASEADO EM SISTEMA MULTIAGENTE

Tese aprovada como requisito parcial para obtenção do grau de Doutor no Curso de Pós-graduação em Informática da Universidade de Brasília, pela Comissão formada pelos professores:

Orientador: Prof. Dr. Maria Emília M. T. Walter – orientadora(CIC/UnB)

Prof. Dr. Célia Ghedini Ralha (CIC/UnB)

Prof. Dr. Marcelo de Macedo Brígido (Bio/UnB)

Prof. Dr. Nalvo Franco de Almeida Junior (UFMS)

Dr. Roberto Coiti Togawa (Embrapa)

Vista e permitida a impressão.
Brasília, 10 de julho de 2015.

Prof.ª Dr.ª Alba Cristina Magalhães Alves de Melo
Programa de Pós-Graduação em Informática
Departamento de Ciência da Computação
Universidade de Brasília

Dedicatória

Ao meu pai, que plantou em mim a semente da curiosidade científica.

À minha mãe, pelo exemplo de força e determinação.

Devo tudo a vocês.

Agradecimentos

É sempre complicado agradecer a todos os que contribuíram para um trabalho desta índole. Com a certeza que muitas outras pessoas ou entidades contribuíram para a realização deste trabalho (e com o meu muito obrigado a esses também), vou tentar agradecer àqueles que de forma mais intensa contribuíram para a realização deste projeto.

Agradeço primeiramente a Deus que me dotou de capacidade, saúde e condições para que chegasse até aqui, por mais essa conquista.

Aristósteles, especificamente a respeito de orientadoras, escreveu: "Este é o ponto essencial: a partir de sua preparação científica, a orientadora deve instigar não apenas a capacidade, mas o desejo de observar fenômenos naturais. No nosso sistema, ela deve tornar-se uma influência passiva, muito mais do que ativa, e sua passividade deve ser composta de uma ansiosa curiosidade científica, e um respeito absoluto pelo fenômeno que ela deseja observar. A orientadora deve entender e sentir sua posição de observadora; a atividade deve permanecer no fenômeno."

Uma orientadora é tanto uma mentora quanto uma fonte de conselhos científicos; ela encoraja o interesse de seus orientados, ao invés de promover seus próprios interesses; ela encontra-se disponível para dar conselhos em sua tese e na sua carreira científica. Uma boa orientadora, além de tudo, sabe ouvir, cientificamente e pessoalmente, seus orientados, compreendendo-nos, mesmo quando sequer tenhamos verbalizado palavra alguma. Ela possui o misterioso dom de perceber e sentir o que se passa em nossas mentes, quando nós mesmos não somos capazes de fazê-lo. Ela é como um farol, no meio das tempestades, que sabemos estar lá para nos orientar; um porto seguro, onde podemos ancorar nossas dúvidas.

Nesse sentido, tive a sorte de conseguir uma orientadora no doutorado, que preencheu não são os requisitos de boa orientadora, como muito mais do que é possível conceber. Eu gostaria de agradecer minha orientadora, Maria Emilia M. T. Walter, por ensinar-me a entender Projeto e Complexidade de Algoritmo e Tópicos em Fundamentos e Métodos de Computação/Bioinformática e a ver muitas coisas sob uma perspectiva diferente. Sua intuição e comentários foram sempre de grande valia, mantendo minha mente fervilhando de pensamentos muitas horas após nossos debates. Nossos debates, sem dúvida alguma, foram um dos pontos mais interessantes e esclarecedores, dos quais terei muita saudade.

Obrigado professora por surpreender-me com sua sagacidade, compreendendo e relevando épocas difíceis pelas quais passei. Em várias situações, ela simplesmente parecia ler meus pensamentos ... Seu senso de justiça e imparcialidade sempre me fascinaram, pois são ponderados com o bom senso.

Gostaria, também, de agradecer meus outros professores ligados ao CIC, Alba Cristina M. A. de Melo, Mauricio Ayala Rincón, Vander Ramos Alves, Marcelo Ladeira e Li Weigang. Obrigado pela dedicação em sala de aula, tal dedicação é digna de louvor e

não será esquecida. Citando Aristóteles, *"In the arena of human life the honors and rewards fall to those who show their good qualities in action"*.

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

Em especial, agradeço a professora Célia Ralha pela atenção e carinho ao revisar os artigos que escrevi, bem como esta tese. Também pela sua confiança, pelos conselhos e direções ao longo de todos esses anos. E acima de tudo, serva do Deus vivo!

Agradeço também ao professor Peter F. Stadler, que me recebeu tão bem na Universidade de Leipzig, Alemanha. Conversar com Peter F. Stadler foi um dos momentos da vida no qual aprendi mais do que em muitos anos de estudo em livros, e nossa conversa motivou-me a modificar a arquitetura do sistema, além de mostrar outras ferramentas interessantes para acoplar ao sistema. Obrigado pelas poucas oportunidades que tive, fica claro que é fundamental para o progresso científico de nosso país trocar experiências com pesquisadores estrangeiros.

Eu quero agradecer enormemente aos meus queridos pais, José da Costa Arruda e Luzia Maria Arruda, pela minha formação de caráter, pelas oportunidades proporcionadas e pela consideração que têm por mim, pela educação, carinho, amor e por muitas vezes que sacrificaram seus sonhos para que os meus fossem realizados. Por me proporcionarem esta oportunidade, pelo apoio incondicional e incentivo para seguir em frente e não desistir no primeiro obstáculo. A parte boa em mim é fruto deles. Agradeço também minha querida irmã Rosângela Arruda, e meus irmãos Wesley Arruda e Warley Arruda. Confesso que a ausência de vocês muito me faz falta.

A todos meus familiares, minhas profundas desculpas pelos momentos que deixei de compartilhar com vocês. Peço apenas que relevem isso por eu estar fazendo algo que muito gosto.

Aos amigos e companheiros de curso Daniel Saad, Breno Moura, Fábio Silva, João Victor e Lucas Ângelo pelo suporte integral, conversas e desabafos, e pela consideração. Obrigada pela acolhida e pela amizade!

Agradeço aos membros da banca examinadora por terem aceitado o convite e pela participação.

À CAPES pelo auxílio financeiro concedido.

Porém, os erros que porventura eu tenha cometido sobre o que escrevi nesta tese são de responsabilidade única e exclusivamente minha; os acertos são méritos de todas essas pessoas.

Em sua tese, Usama Fayyad escreveu que ele não sabia por que as pessoas exageram agradecendo "quase todas as pessoas no planeta". Sem exageros, tenho certeza que agradeci apenas uma pequena fração das pessoas a quem devo muito e peço especial perdão àquelas que eu omiti inconscientemente.

Obrigado a todos!

Resumo

Os RNAs não-codificadores (ncRNAs) constituem um importante subconjunto dos transcritos produzidos nas células dos organismos, pois afetam diversos processos celulares. Embora existam métodos computacionais bastante eficazes para identificar proteínas, a anotação de ncRNAs é hoje objeto de pesquisa intensa, pois suas características e sinais não são ainda completamente conhecidos. Neste contexto, nesta tese, apresentamos uma arquitetura para anotação de ncRNAs baseada no paradigma de Sistema Multiagente. A implementação do sistema, denominado de ncRNA-Agents, usa agentes colaborativos, em que cada agente tem conhecimento e raciocínio (simulando os de biólogos) sobre um aspecto específico de RNA, o que contribui para uma anotação curada de ncRNA, com qualidade associada e explicações baseadas nos resultados das ferramentas usadas pelo sistema para recomendar a anotação. Além disso, foram realizados três estudos de casos com os fungos *Saccharomyces cerevisiae*, *Paracoccidioides brasilienses* e *Schizosaccharomyces pombe*, para avaliar o desempenho do sistema quanto a sua capacidade de anotar ncRNAs conhecidos e de predizer novos ncRNAs. Acesso público a esta ferramenta está em http://www.biomol.unb.br/ncrna-agents.

Palavras-chave: RNAs não-codificadores, Anotação de RNAs não-codificadores, Sistema Multiagente, Bioinformática, Inteligência Artificial

Abstract

Non-coding RNAs (ncRNAs) are an important subset of the transcripts produced in the cells of organisms, since they affect many cellular processes. Although there are efficient and fast computational methods to identify proteins, annotation of ncRNAs is now focus of intensive research once their characteristics and signals are not yet entirely known. In this context, in this thesis, we present an architecture for ncRNAs annotation based on the multi-agent system paradigm. The implementation of a system, called ncRNA-Agents, uses collaborative agents, where each agent has knowledge and reasonig (simulating biologists) about a specific aspect of RNA, which contributes to a curated ncRNA annotation, with associated quality and explanations based on the results of the tools used by the system to recommend the annotation. In addition, we performed three case studies with three fungi, *Saccharomyces cerevisiae*, *Schizosaccharomyces pombe* and *Paracoccidioides brasiliensis*, to evaluate the performance of the system and its ability to annotate known ncRNAs and predict new ncRNAs. This tool is publicly available at http://www.biomol.unb.br/ncrna-agents.

Keywords: non-coding RNA, Annotation of Non-Coding RNAs, Multi-agent System, Bioinformatics, Artificial Intelligence.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Desde o trabalho de Watson e Crick [150], em que eles propuseram a estrutura em dupla hélice para uma molécula de DNA, a comunidade científica vem realizando grandes esforços para tentar compreender a estrutura e o funcionamento da biologia molecular dos seres vivos. Na década de 1990, iniciou-se um consórcio internacional, que teve o intuito de mapear e sequenciar o genoma humano por completo. Concluído em 2001, esse projeto sequenciou o genoma humano com 3 bilhões de bases e de 20.000 a 30.000 genes [72, 132, 144].

Diferentes técnicas de sequenciamento de genomas foram sendo desenvolvidas desde o Projeto Genoma Humano, e atualmente os sequenciadores de alto desempenho geram um enorme volume de dados, que precisam de métodos computacionais e algoritmos sofisticados que suportam a extração de conhecimentos biológico dessa expressiva massa de informações.

Neste contexto, a Bioinformática utiliza conhecimentos das áreas da Computação, Matemática e Estatística, com a finalidade de resolver problemas de Biologia Molecular. Nessa área, são desenvolvidos *pipelines* e ferramentas computacionais para apoiar os biólogos em projetos de sequenciamento de genomas de modo a converter dados experimentais em informações biologicamente relevantes [117, 127, 128]. Nesses projetos, a complexidade dos problemas de Biologia Molecular, além do enorme volume de dados gerados de porções de DNA e de RNA, requerem técnicas sofisticadas de Computação, dentre outras áreas.

Em particular, até a década de 1990, o conhecimento sobre as moléculas de ácidos ribonucléicos (RNAs) estava relacionado basicamente ao uso da informação contida no DNA para a tradução de proteínas, como o RNA mensageiro (mRNA), o RNA transportador (tRNA) e o RNA ribossomal (rRNA) [132]. Porém, desde então, descobriram-se outros tipos de moléculas de RNA, que não são traduzidas em proteínas, mas estão presentes nos organismos, afetando uma grande variedade de processos celulares. Essas moléculas, antes chamadas de lixo, são hoje denominadas de RNAs não-codificadores (ncRNAs) [77].

Os ncRNAs controlam uma gama notável de reações biológicas e processos, como iniciação da tradução, controle da abundância de mRNA, arquitetura do cromossomo, manutenção de células-tronco, desenvolvimento do cérebro, músculos e secreção de insulina, dentre outras [95].

Apesar de sua importância funcional, e de muitas pesquisas buscarem classicar e identificar ncRNAs, os métodos biológicos e computacionais ainda não são capazes de

identificá-los e classificá-los, o que afeta diretamente a anotação de ncRNAs.

Do ponto de vista experimental, os ncRNAs são caracterizados pela ausência de tradução em proteínas. Do ponto de vista computacional, o fato de sequências de certas classes de ncRNAs serem curtas e não terem um padrão de sequência bem comportado impedem que sejam reconhecidos apenas pelas suas bases (sequências primárias), o que significa que em geral devem ser caracterizadas pelas suas estruturas secundárias (conformação espacial) [62]. Uma observação importante é a de que, em geral, ncRNAs não podem ser identificados e classificados pelas mesmas ferramentas que detectam genes codificadores de proteína de forma eficiente, como o BLAST [120].

Por outro lado, Sistemas Multiagentes (SMAs), dentro de Inteligência Artificial, são caracterizados pela distribuição da inteligência entre diferentes entidades autônomas (agentes), que interagem para atingir objetivos individuais ou coletivos. Para tanto, os agentes que compõem um SMA precisam negociar, cooperar para atingir objetivos (que não podem ser realizados por um só agente) e coordenar esforços conjuntos [157].

Este trabalho propõe o uso de um SMA para recomendar anotação para um ncRNA, particularmente utilizando ferramentas baseadas em técnicas de raciocínio automatizado.

1.1 Motivação

Identificar e anotar ncRNAs constituem-se hoje em pesquisas desafiadoras tanto em Biologia Molecular, quanto em Bioinformática, devido a descobertas recentes de que ncRNAs exercem funções diversificadas e importantes nos mecanismos celulares, como a regulação do metabolismo de outras moléculas, o auxílio do transporte de proteína, a edição de nucleotídeos, a regulação de *imprinting* e estado da cromatina [131]. A tendência atual de pesquisa é usar várias ferramentas diferentes e, com base nesses resultados, os biólogos usarem seu raciocínio biológico para decidirem a anotação de sequências que potencialmente seriam ncRNAs.

1.2 Problema

Tanto quanto sabemos, não há ferramenta computacional usando simulação do raciocínio dos biólogos para, a partir do resultado de diversas ferramentas, recomendar anotação de ncRNAs.

1.3 Objetivos

O objetivo principal deste trabalho é propor um SMA para a anotação de ncRNAs, denominado ncRNA-Agents, utilizando diversas ferramentas e bancos de dados conhecidos e regras de inferência para simular o raciocínio dos biólogos.

Como objetivos específicos pode-se citar:

- Propor uma arquitetura baseada em SMA para anotação de ncRNAs;
- Propor uma ferramenta baseada na arquitetura proposta, e disponibilizar pela web;
- Realizar testes com dados reais de projetos de sequenciamento de genomas;

- Comparar a ferramenta com outras existentes na literatura.

1.4 Descrições dos Capítulos

No Capítulo 2, serão abordados conceitos relativos a ncRNAs, por meio da apresentação de algumas classes conhecidas. Serão mostrados repositórios (base de dados) e ferramentas computacionais comumente usadas para anotar ncRNAs. Por fim, serão discutidos os desafios para e anotação de ncRNAs, que motivaram o desenvolvimento desta pesquisa.

No Capítulo 3, serão descritos conceitos básicos e propriedades de SMAs, em particular ferramentas que permitem implementar sistemas baseados em agentes. Além disso, são apresentadas ferramentas que simulam raciocínio. Avaliações feitas dessas ferramentas justificam as escolhas realizadas para a implementação do SMA.

No Capítulo 4, propomos a arquitetura do ncRNA-Agents, um SMA para anotar ncRNAS. Serão detalhados o sistema implementado, além das ferramentas e bancos de dados usados.

No Capítulo 5, discutimos os resultados obtidos para três fungos, anotados pelo sistema ncRNA-Agents.

Finalmente, no Capítulo 6, são apresentadas as conclusões e sugeridos trabalhos futuros.

Capítulo 2

RNAs não-codificadores

Neste capítulo serão descritos RNAs não-codificadores (ncRNAs) e seus papéis nos mecanismos celulares, necessários ao entendimento deste trabalho. Na Seção 2.1 são apresentados os ácidos nucléicos, que contêm informações e mecanismos para sintetizar proteínas, além de uma breve descrição do processo de síntese de proteínas, através do qual as informações contidas no DNA, e diversos tipos de RNAs, são utilizados para produzir proteínas. Na Seção 2.2, detalhamos diferentes tipos de ncRNAs, ferramentas computacionais para anotação de ncRNAs e bancos de dados de ncRNAs. Na Seção 2.3 seguem características e desafios na anotação de ncRNAs, que serviram de base para o sistema de anotação proposto nesta tese.

2.1 Ácidos Nucléicos e Síntese de Proteínas

Os ácidos nucléicos são polímeros formados a partir de moléculas mais simples, chamadas de nucleotídeos. Um nucleotídeo tem na sua composição uma molécula de açúcar com cinco átomos de carbono (pentose), ligada a um grupo fosfato e uma base nitrogenada [69], como podemos ver na Figura 2.1.

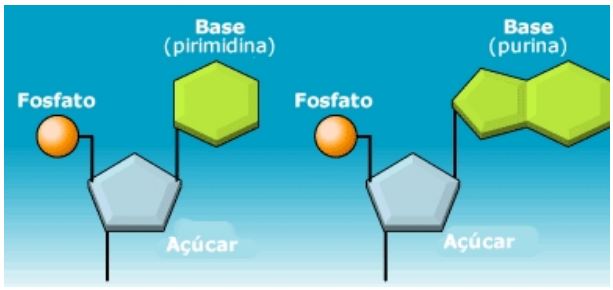


Figura 2.1: Esquema da estrutura de um nucleotídeo, mostrando seus principais componentes: açúcar, fosfato e base nitrogenada [80].

Os ácidos nucleicos podem ser de dois tipos: o **ácido desoxirribonucleico** (do inglês, *deoxyribonucleic acid*), conhecido por DNA, e o **ácido ribonucleico** (do inglês, *ribonucleic acid*), conhecido por RNA. O DNA e o RNA possuem diferenças tanto composicionais quanto estruturais, que serão descritas em seguida.

Dentre as várias funções dos ácidos nucléicos está a de codificar os aminoácidos que compõem as proteínas. Essa codificação é a base do processo de síntese de proteínas, explicado na seção 2.1.3.

2.1.1 DNA

O ácido desoxirribonucléico (DNA) contém as informações genéticas de um organismo vivo [133]. Nos organismos procariotos (cujas células não contêm núcleos), o material genético está espalhado na célula. Nos organismos eucariotos (cujas células contêm núcleos), o DNA está localizado no núcleo, e contém informações sobre como, quando e onde produzir cada tipo de proteína [80].

As bases nitrogenadas, que constituem os nucleotídeos podem ser divididas em dois grupos: purinas ou pirimidinas. As bases purinas são a citosina (C) e a timina (T), e as pirimidinas incluem a Adenina (A) e a Guanina (G). Assim, existem quatro tipos de nucleotídeos, de acordo com sua base nitrogenada (Figura 2.2).

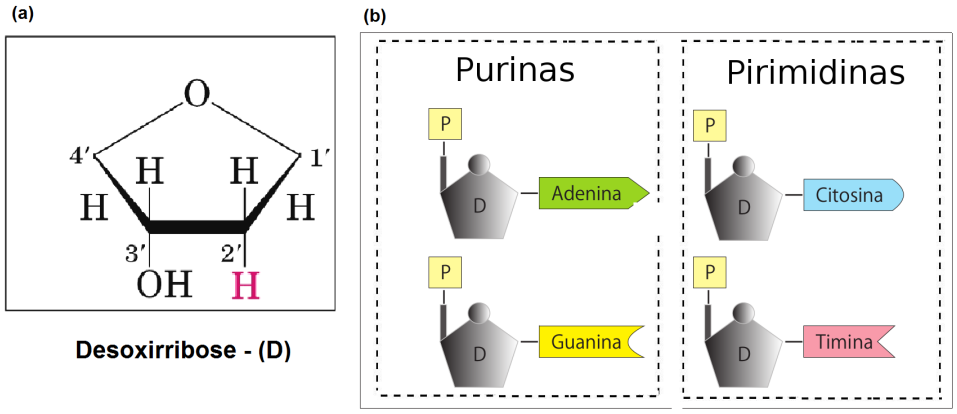


Figura 2.2: (a) O açúcar desoxirribose do DNA. (b) Os quatros tipos de nucleotídeos que compõem a molécula de DNA, onde P representa o grupo fosfato, D o açúcar do DNA (desoxirribose), e as bases nitrogenadas, que podem ser A, G, C ou T (adaptado de [80]).

A estrutura do DNA é formada por duas cadeias ou fitas paralelas compostas por uma sequência de nucleotídeos, que são unidas através de pontes de hidrogênio criadas entre as bases nitrogenadas de cada fita, onde a base A estará pareada com a base T (A-T) e C com G (C-G) [138]. As bases A-T e C-G são chamadas de bases complementares. Essas duas fitas estão dispostas em espiral em torno de um eixo. Cada fita possui uma extremidade chamada 5' e a outra chamada 3', o que cria uma orientação em cada uma das fitas, denominadas de senso e antisenso, respectivamente. As duas cadeias ficam, portanto em direção antiparalelas (opostas), formando uma dupla hélice (Figura 2.3).

O DNA pode sofrer replicações através de um processo conhecido como duplicação semiconservativa, onde cada molécula de DNA recém formada possui uma porção das cadeias da molécula mãe [81]. Para realizar a replicação, a dupla fita do DNA abre-se, através do rompimento das pontes de hidrogênio, e nucleotídeos livres encaixam-se na molécula através de novas pontes de hidrogênio. Os nucleotídeos vão sendo ligados

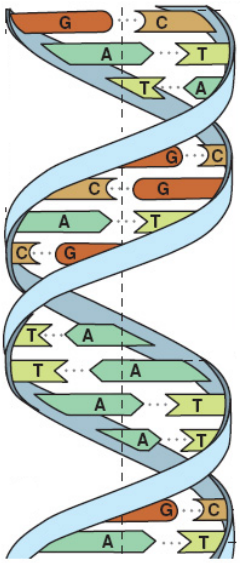


Figura 2.3: Dupla hélice do DNA, indicando as ligações entre as bases complementares [2].

entre si pela enzima DNA polimerase. O resultado desse processo é a formação de duas moléculas de DNA idênticas à original (Figura 2.4).

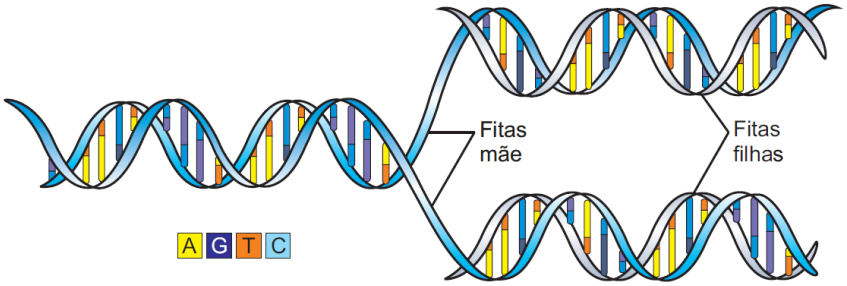


Figura 2.4: À esquerda podemos observar um trecho de uma molécula de DNA que evidencia o aspecto de dupla hélice e à direita as duas fitas mães separadas, onde cada fita serve de molde, na replicação, para as fitas filhas, que formam duas moléculas semelhantes à dupla hélice original [80].

2.1.2 RNA

O RNA é uma molécula que tem estrutura semelhante à do DNA, exceto pela presença da ribose ao invés da desoxirribose e da base uracila no lugar da timina, além do fato da molécula de RNA ser formada por apenas uma cadeia de nucleotídeos (Figura 2.5).

O RNA é também formado pelo grupo fosfato, açúcar (ribose) e por uma base nitrogenada. Porém, entre as bases nitrogenadas, a Uracila (U) substitui a Timina (T) do DNA [80], sendo U uma base purina (Figura 2.5).

Existem três tipos principais de RNA: o RNA mensageiro (mRNA), o RNA ribossomal (rRNA) e o RNA transportador (tRNA), que atuam no processo de síntese de proteínas

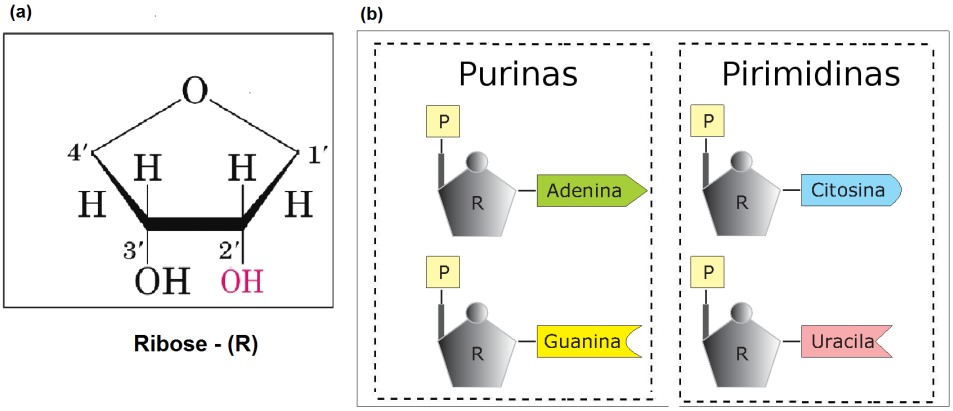


Figura 2.5: (a) O açúcar ribose do RNA. (b) Os quatros tipos de nucleotídeos que compõem a molécula de RNA, observando-se que a Uracila (U) substitui a Timina (T) do DNA, P representa o grupo fosfato, R a ribose e as bases nitrogenadas A, G, C e U.

.

(Figura 2.6). De forma resumida, o mRNA é responsável por carregar a informação genética, copiada do DNA, que estabelecerá a sequência dos aminoácidos na proteína; o rRNA é o principal componente dos ribossomos, local onde ocorre de fato a síntese de proteínas; e, por fim, o tRNA identifica e carrega os aminoácidos para os ribossomos. Porém, existem outros tipos de RNAs, que serão descritos na seção 2.2.

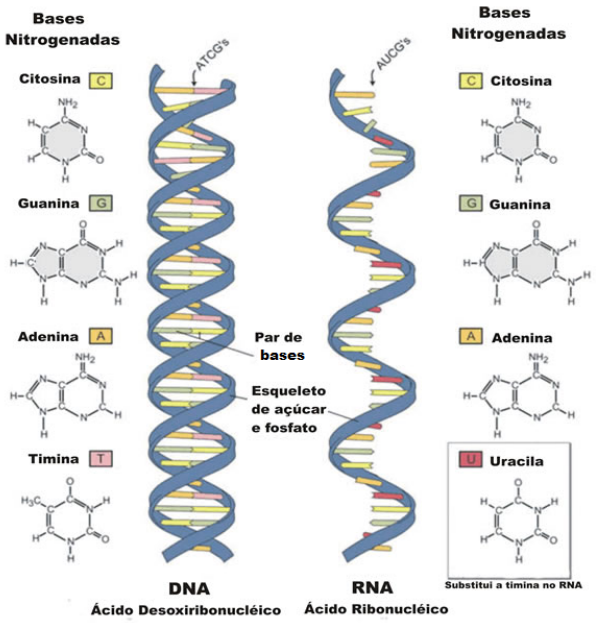


Figura 2.6: RNA e suas bases nitrogenadas à esquerda e DNA à direita (Adaptado de [23]).

2.1.3 Síntese de Proteínas

Nesta seção, serão descritos como as informações presentes em uma molécula de DNA são utilizadas na síntese de proteína, processo conhecido como Dogma Central da Biologia

Molecular.

Em primeiro lugar, a transcrição é sintetizada pela enzima RNA polimerase, que vai ligar-se a uma determinada sequência de nucleotídeos do DNA, identificada pelas regiões promotoras[1], e percorre a fita de DNA (fita molde) até encontrar as regiões terminadoras[2]. A transcrição baseia-se no pareamento de bases complementares, usando uma das fitas do DNA como molde, ou seja, adenina com uracila (A → U), timina com adenina (T → A) e citosina com guanina (C ↔ G) e vice-versa. A molécula de RNA sintetizada a partir de uma região do DNA é o mRNA [80].

O processo de transcrição acontece tanto nos procariotos (Figura 2.7) como nos eucariotos, quem têm um processo de transcrição um pouco mais complexo que os procariotos.

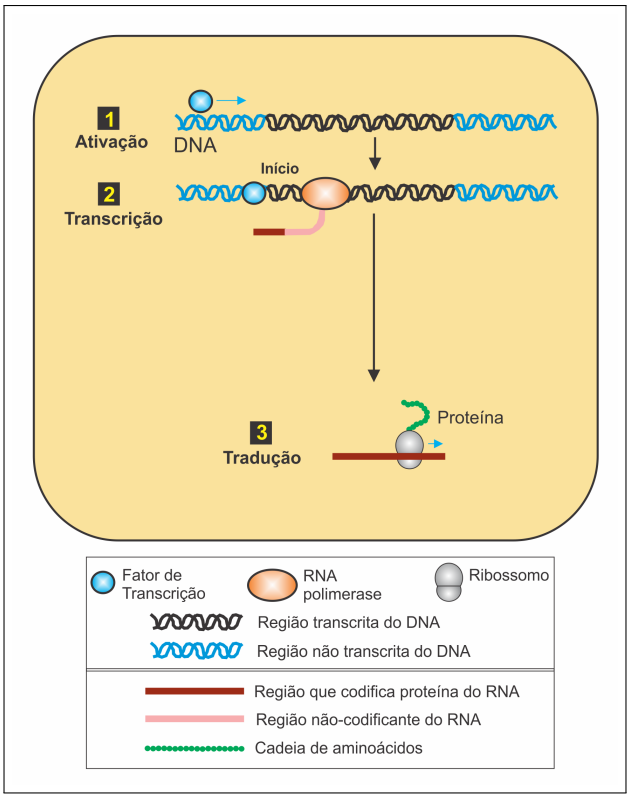


Figura 2.7: A síntese protéica em célula procariótica funciona da seguinte maneira. No passo 1, fatores de ativação ligam-se às regiões da regulação dos genes que controlam, ativando-os. No passo 2, a RNA polimerase começa a transcrição do gene ativado pela região promotora, resultando na formação do mRNA. No passo 3, o mRNA é ligado ao ribossomos. Nesse ponto, a proteína é sintetizada pelo ribossomo, que liga os aminoácidos em uma cadeia linear [80].

Todos os organismos possuem maneiras de controlar quando os seus genes podem ser transcritos [80]. Muitas células são capazes de responder a sinais externos ou a alterações das condições externas, "ligando" ou "desligando" genes específicos. Dessa forma, as células adaptam-se às necessidades do ambiente.

[1] É aquela onde se associam proteínas como fatores de transcrição e também a enzima RNA polimerase [47].

[2] É aquela em que se pode observar o códon de término de tradução e transcrição [47].

Nos eucariotos, o mRNA recém-trancrito é conhecido como pré-mRNA, e em alguns organismos, ele irá sofrer algumas modificações antes que se transforme em um mRNA maduro [133]. No decorrer desse processo de maturação ocorre o *splicing* (eliminação dos íntrons do pré-mRNA).

O mRNA formado, através da transcrição, move-se para o citoplasma, precisamente para os ribossomos, onde ocorre a segunda parte do processo para a síntese da proteína: a tradução. Cada aminoácido é codificado por um grupo de três bases do DNA, que recebe o nome de *códon*. No processo de tradução, primeiramente o mRNA liga-se entre as duas subunidades do ribossomo, onde cada códon do mRNA é pareado com o anticódon correspondente que está presente em moléculas de tRNA [133]. A Figura 2.8 mostra a síntese de proteína em eucariotos [80, 85].

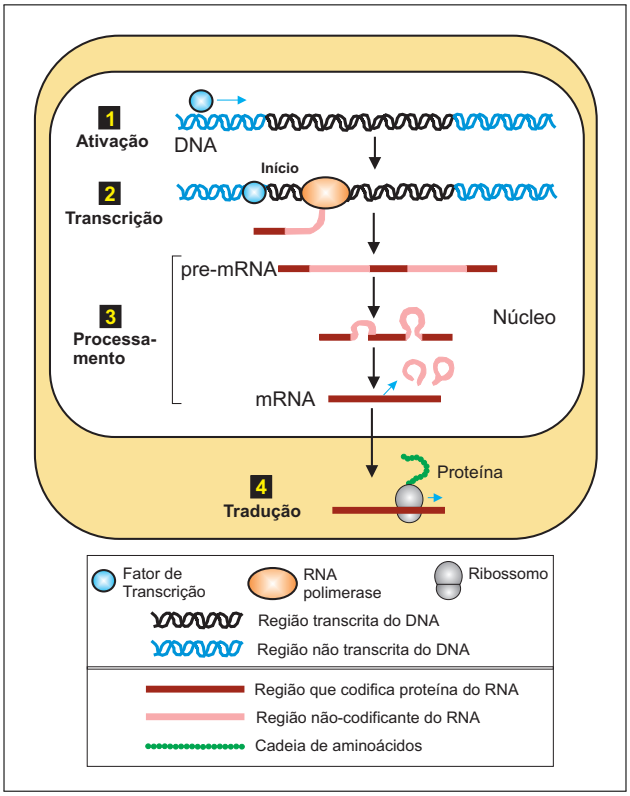


Figura 2.8: A síntese protéica em célula eucariótica funciona da seguinte maneira. No passo 1, fatores de ativação ligam-se às regiões da regulação dos genes que controlam, ativando-os. No passo 2, a RNA polimerase começa a transcrição do gene ativado pela região promotora, resultando na formação do pré-mRNA. No passo 3, é feito um processamento na transcrição para remover sequências não-codificadora (íntrons). Para finalizar, no passo 4, o mRNA move-se para o citoplasma ligando ao ribossomo. Nesse ponto, a proteína é sintetizada, ligando dos aminoácidos em uma cadeia linear [80].

Os pareamentos dos tRNAs são feitos em sequência, e à medida que os aminoácidos vão sendo ligados, os tRNAs são liberados. Esse processo é repetido até que apareça um sinal de terminação no mRNA. Então, mRNAs são RNAs que geram proteínas. Cada códon corresponde a um único aminoácido, porém um mesmo aminoácido pode ser definido por mais de um códon (Tabela 2.1) [81, 133]. Existem ainda três códons (UAG, UAA e

UGA) que não correspondem a nenhum aminoácido, mas indicam sinais de término da tradução [133]. A Tabela 2.1 mostra a correspondências entre códons e aminoácidos.

Tabela 2.1: O Código Genético [145].

Primeira posição	Primeira posição				Terceira posição
	U	C	A	G	
U	UUU Phe	UCU Ser	UAU Tyr	UGU Cys	U
	UUC Phe	UCC Ser	UAC Tyr	UGC Cys	C
	UUA Leu	UCA Ser	UAA Stop	UGA Stop	A
	UUG Leu	UCG Ser	UAG Stop	UGG Trp	G
C	CUU Leu	CCU Pro	CAU His	CGU Arg	U
	CUC Leu	CCC Pro	CAC His	CGC Arg	C
	CUA Leu	CCA Pro	CAA Gln	CGA Arg	A
	CUG Leu	CCG Pro	CAG Gln	CGG Arg	G
A	AUU Ile	ACU Thr	AAU Asn	AGU Ser	U
	AUC Ile	ACC Thr	AAC Asn	AGC Ser	C
	AUA Ile	ACA Thr	AAA Lys	AGA Arg	A
	AUG Met[a]	ACG Thr	AAG Lys	AGG Arg	G
G	GUU Val	GCU Ala	GAU Asp	GGU Gly	U
	GUC Val	GCC Ala	GAC Asp	GGC Gly	C
	GUA Val	GCA Ala	GAA Glu	GGA Gly	A
	GUG Val	GCG Ala	GAG Glu	GGG Gly	G

[a] AUG faz parte do sinal de inicialização.

Na natureza, são catalogados vários aminoácidos [75], sendo 20 deles comumente encontrados em proteínas. A Tabela 2.2 mostra o nome, a abreviação e o código de uma letra usada para identificar cada um dos 20 aminoácidos.

Os aminoácidos unidos por ligações peptídicas constituem as proteínas. Elas estão envolvidas em todos os processos biológicos dos seres vivos, como nas funções estruturais, catalisadoras e reguladoras [78]. Além disso, as proteínas participam direta ou indiretamente de quase todas as atividades celulares de um organismo vivo [132].

2.1.4 Genes e regiões funcionais

Gene, no contexto do processo de síntese de proteínas, corresponde a um fragmento de DNA que pode ser transcrito em um pré-mRNA. As regiões do DNA situadas entre os genes são chamadas de regiões intergênicas.

Como dito antes, nem toda a informação de um gene é utilizada para a produção de proteínas e parte do pré-mRNA é descartado no processo de *splicing*. Com base nessa informação, os genes contêm partes denominadas éxons e outras denominadas íntrons. Um éxon é um trecho contíguo de uma sequência de DNA que vai ser utilizado na síntese do mRNA. Um íntron é um trecho do DNA que é descartado no processo de *splicing*. De acordo com a posição onde se encontram dentro do gene, os éxons podem ser classificados

Tabela 2.2: Lista dos aminoácidos encontrados na natureza

Nome	Abreviação	Código
Alanina	Ala	A
Arginina	Arg	R
Asparagina	Asn	N
Ácido Aspártico	Asp	D
Cisteína	Cys	C
Glutamina	Gln	Q
Ácido Glutâmico	Glu	E
Glicina	Gly	G
Histidina	His	H
Isoleucina	Ile	I
Leucina	Leu	L
Lisina	Lys	K
Metionina	Met	M
Fenilalanina	Phe	F
Prolina	Pro	P
Serina	Ser	S
Treonina	Thr	T
Triptofano	Trp	W
Tirosina	Tyr	Y
Valina	Val	V

em quatro classes: éxon inicial (primeiro éxon do gene), éxon final (último éxon do gene), éxon interno (qualquer éxon situado entre os éxons inicial e final) e éxon único (éxon componente de um gene constituído por um único éxon). As regiões correspondentes aos éxons de uma sequência de DNA, também são chamadas de regiões codificadoras.

Os genes podem codificar mais de uma proteína devido ao processo chamado *splicing* alternativo, onde vários mRNAs maduros (mRNAs obtidos após *splicing* dos íntrons) diferentes podem ser sintetizados a partir de um mesmo gene, utilizando subconjuntos distintos do conjunto original de éxons. Existem outras porções de sequências de DNA com papéis variados na expressão gênica, além dos éxons e dos introns. Essas regiões são conhecidas como regiões funcionais. Abaixo segue uma lista com algumas delas:

- **região promotora**: localiza-se no início de um gene. A enzima RNA-polimerase liga-se a esta região para dar início à transcrição;
- **região terminadora**: localiza-se no final de um gene e sinaliza o final do processo de transcrição.

Um exemplo de fita de DNA com os éxons, íntrons e algumas regiões funcionais pode ser visto na Figura 2.9.

2.2 Tipos de RNAs não-codificadores (ncRNAs)

Apesar de identificados, e possuírem papel de grande importância, a caracterização dos RNAs não-codificadores (ncRNAs) foi, nas décadas de 1980 e 1990, relegada para segundo plano, talvez devido a dificuldades técnicas relacionadas a identificar essas moléculas pequenas, instáveis e pouco abundantes [36]. Nesta época, os RNAs não envolvidos diretamente com a síntese de proteínas eram chamados de DNA lixo (*junk* DNA) [132].

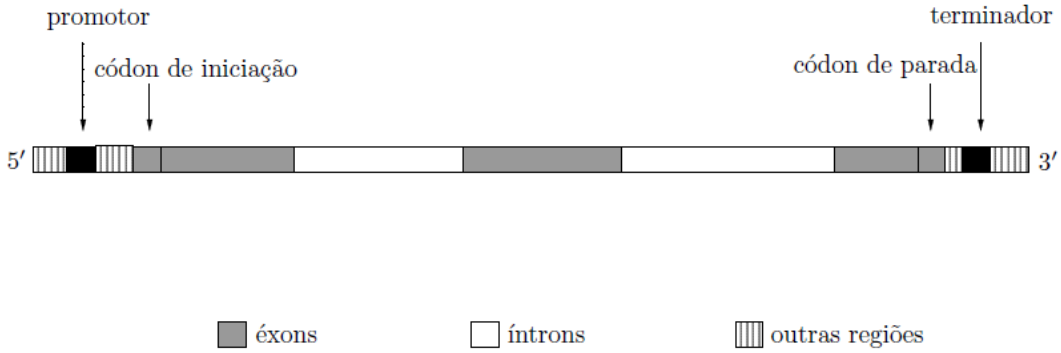


Figura 2.9: Exemplo de uma fita do DNA com um gene composto por três éxons.

A partir do início dos anos 2000, retomaram-se os estudos dos ncRNAs, em consequência da crescente quantidade de ncRNAs identificados pelos biólogos, descritos na literatura. As descobertas mais notáveis envolvendo RNAs estruturais estão relacionadas ao desenvolvimento do sistema nervoso, colaborando com a observação de que a quantidade de regiões não-codificadoras é proporcional à complexidade dos organismos [7, 89, 94, 116]. Usando *small interfering* RNA (siRNA) e microRNA (miRNA) encontrados nos organismos, foram desenvolvidos novos métodos baseados nos mecanismos nos quais esses RNAs estavam envolvidos, citando principalmente silenciamento de RNAs mensageiros alvo [118].

Por outro lado, o primeiro enunciado do Dogma Central da Biologia Molecular dizia que a função dos RNAs restringia-se a participação em síntese de proteínas. Contudo, estudos posteriores revelaram que cerca de 98% do que é transcrito pelo genoma humano é constituído de ncRNAs [88]. Uma quantidade tão grande de ncRNAs certamente desempenha papeis importantes nos organismos.

De forma genérica, existem duas grandes classes de RNAs, os codificadores de proteínas e os não-codificadores de proteínas, como podemos observar na Figura 2.10.

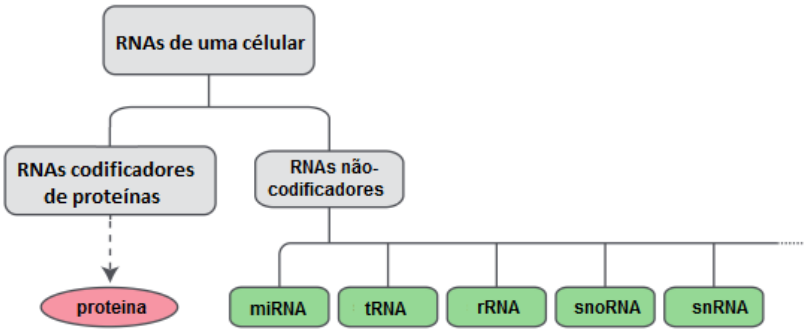


Figura 2.10: Os RNAs de uma célula apresentam duas grandes classes, os que codificam proteínas e os que não codificam proteínas (adaptada de [20]).

Além disso, mostrou-se que os ncRNAs participam de quase todo processo de regulação da expressão gênica [3, 26, 110]. NcRNAs controlam a expressão gênica atuando em eventos transcricionais, pós-transcricionais e traducionais. Como podemos ver na Figura 2.11,

as funções reguladoras de ncRNAs estão envolvidos em todas as etapas do Dogma Central da Biologia Molecular, na transcrição, pós-transcrição e tradução. A informação genética flui do DNA (genótipo) para o RNA (com linhas pretas e grossas). O RNA (fenótipo) codifica proteínas e/ou ncRNAs a partir de mRNAs. Os ncRNAs controlam a regulação da expressão gênica (como podemos ver nas linhas tracejadas), ao passo que ncRNAs estruturais (por exemplo, rRNA e tRNA) estão envolvidos na síntese de proteínas [84].

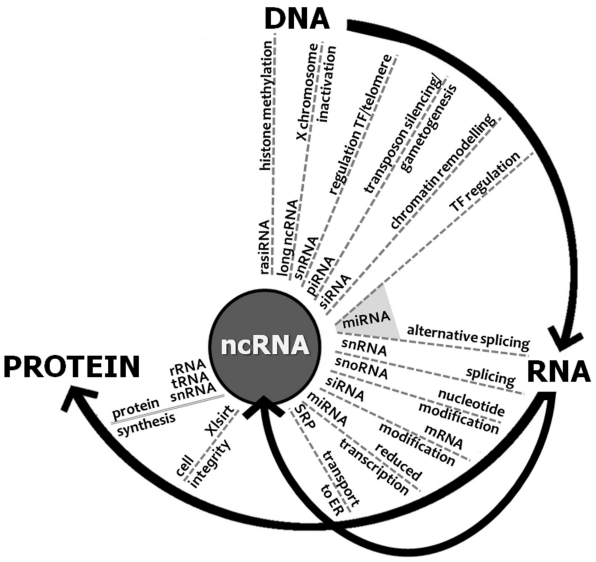


Figura 2.11: Funções reguladoras de ncRNAs [27].

Mais ainda, diferentes classes de ncRNAs foram relatadas em processos de regulação tais como: pequenos RNAs. Como snoRNAs, miRNAs, siRNAs e piRNAs [16, 32, 36, 92] e por último os ncRNAs longos (*long non-coding RNAs* - lncRNAs), que incluem exemplos como Xist, Evf, Air, CTN e PINK [25, 44, 70, 97, 104, 115].

Desde a descoberta dos ncRNAs, muitas perguntas têm sido feitas e muitos estudos têm sido direcionados à procura de respostas. Porém, essas moléculas ainda não são bem conhecidas. A principal causa disso é que uma parte grande das pesquisas para detecção de genes, durante muito tempo, foi voltada na direção de RNAs mensageiros e proteínas.

2.2.1 Estruturas primária, secundária e terciária

Nos ncRNAs, pareamentos entre as bases formam componentes estruturais conhecidos como (Figura 2.12):

- talo (*stem*): possui pares de bases complementares;
- alça (*loop*): local onde as bases não são pareadas;

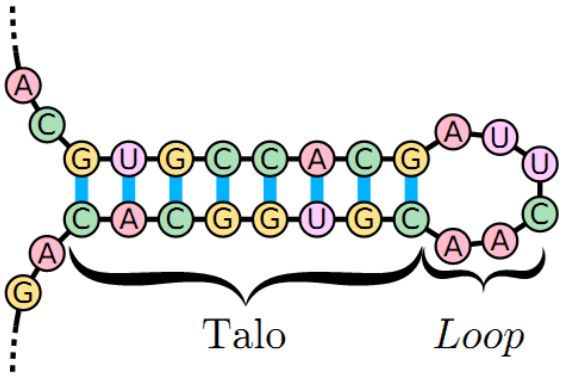


Figura 2.12: Talo - Alça (*Stem*) - (*Loop*) correspondente à estrutura primária ACGUGCCACGAUUCAACGUGGCACAG(Imagem adaptada de [67]).

Na Figura 2.13 podemos observar o dobramento típico da estrutura de um tRNA, o qual apresenta o anti-códon, que é necessário para se ligar à posição correta do mRNA.

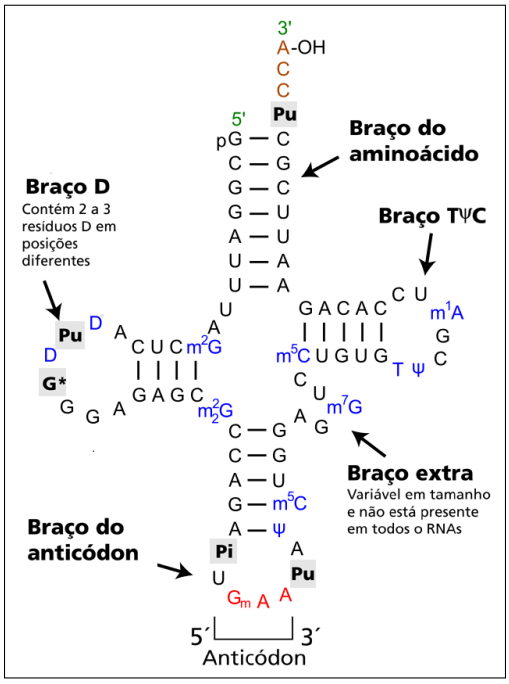


Figura 2.13: Estrutura de um RNA transportador (tRNA), onde os círculos fechados representam os nucleotídeos derivados das bases A, C, G e T; as linhas entre eles representam pontes de hidrogênio, que caracterizam os pares de bases. As características e/ou resíduos estão representados em cinza. Podemos observar os quatro braços sempre presentes e um braço extra que não está presente em todos os tRNAs. Nesta Figura, Pu significa qualquer purina; Pi qualquer pirimidina; G* – guanosina [112].

São criadas várias abstrações para tratar com mais facilidade a estrutura das moléculas de ncRNAs. Existem três formas de representar essas estruturas (Figura 2.14):

- Primária: a sequência de bases (A,T ou U, C e G) define a molécula, sendo essa a estrutura que os sequenciadores automáticos produzem;

- Secundária: corresponde às ligações feitas entre os pares de bases, podendo ser representada em forma espacial 2D;
- Terciária: É uma forma de mostrar a estrutura em formato 3D.

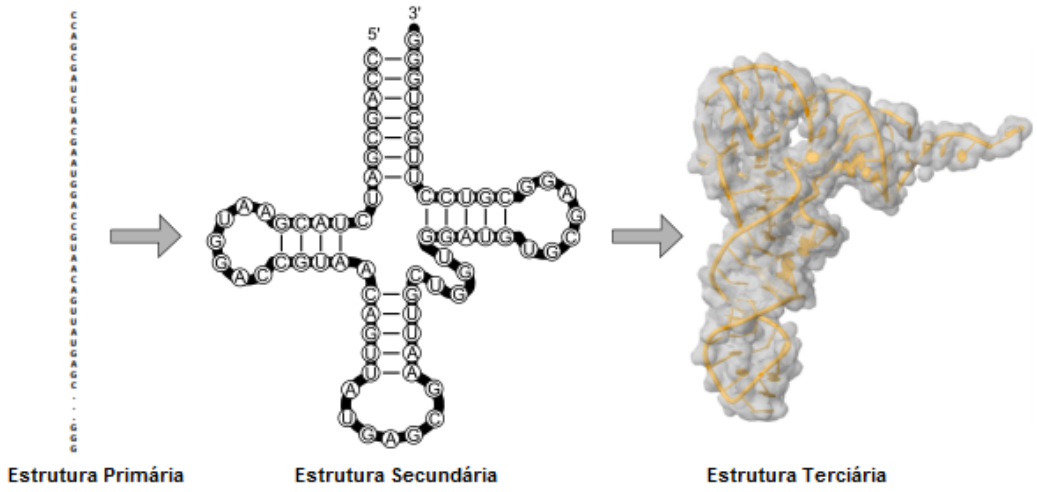


Figura 2.14: Estruturas primária, secundária e terciária. A formação de pares de bases entre as regiões complementares formam uma rede de talos e alças, relativas à estrutura secundária. A estrutura secundária é energeticamente mais estável do que a estrutura terciária [140].

2.2.2 Classificações de ncRNAs

Diferentes classes de ncRNAs são distinguidas pelas suas funções, que dependem diretamente da estrutura e comprimento das suas moléculas, e da composição da suas sequências.

Os ncRNAs tendem a dobrar-se de formas diferentes em suas estruturas secundárias, em parte, porque os RNAs precisam dessa estrutura para serem funcionais, notando-se que essas estruturas são pequenas [112]. O exemplo provavelmente mais conhecido da classe que precisa de uma estrutura para ser funcional são os RNAs transportadores (tRNAs).

A classificação de ncRNAs é feita, em geral, de acordo com suas funcionalidades, o que permite entender o papel que os ncRNAs realizam nos mecanismos celulares. O número de famílias atualmente conhecidas já ultrapassa 2.450 (versão 12.0 do banco de dados Rfam) [22, 49].

As funções da maior parte dos ncRNAs recentemente descobertos não são conhecidas. Abordo brevemente as principais classes de ncRNAs com foco nos ncRNAs "clássicos", conhecidos há algum tempo e, portanto, mais bem caracterizados.

Deve-se ressaltar que, nos últimos anos, as famílias de pequenos ncRNAs e longos ncRNAs vêm crescendo muito e os pesquisadores tentam ainda compreender plenamente suas funcionalidades [20].

2.2.2.1 ncRNAs pequenos

O **RNA transportador (tRNA)** [112] é utilizado como molécula transportadora de informação de cada códon componente do mRNA em um aminoácido específico, a ser adicionado à proteína sendo formada. O tRNA desempenha essa função através de duas regiões, o anticódon que é responsável pelo reconhecimento de códons específicos do mRNA, e o aminoácido correspondente ao códon (Figura 2.13).

O **RNA ribossomal (rRNA)** [36] é o componente central do ribossomo. A função do rRNA é prover um mecanismo para decodificar o mRNA em aminoácidos e interagir com os tRNAs durante a tradução, na síntese de proteína.

O ***small nuclear* RNA (snRNA)** [73, 138] é encontrado no núcleo de uma célula. Eles estão envolvidos no processo de *splicing* do pré-mRNA, em que os íntrons de um transcrito primário são eliminados, resultando no mRNA maduro. A estrutura secundária desses RNAs é altamente conservada nos organismos. Alguns deles, conhecidos como U1, U2, U4, U5 e U6, são essenciais para o *splicing* do pre-mRNA.

O ***small nucleolar* RNA (snoRNA)** [32] é uma classe de pequenas moléculas que realizam modificações químicas no rRNA, e em outros ncRNAs, como o tRNA. Essas modificações possuem como principal objetivo promover a maturação desses ncRNAs, transformando-os em moléculas ativas. A origem desses genes ainda não está clara, mas acredita-se que eles originam-se dos introns do mRNA. Existem basicamente duas classes de snoRNAs, o C/D box e o H/ACA box.

O **microRNA (miRNA)** [92] parece estar relacionado com a regulação gênica. As moléculas de miRNAs são parcialmente complementares a uma ou mais moléculas de mRNA e sua principal função é reduzir a expressão de genes codificadores, inibindo a tradução de mRNAs.

O ***small interfering* RNA(siRNA)** [36] possui o mesmo papel do miRNA, porém reduz a expressão de genes codificadores degradando o mRNA em vez de inibir sua tradução.

O ***piwi-interacting* RNA (piRNA)** [16] é uma classe de pequenas moléculas de RNA existentes basicamente nas células de mamíferos. Assim como os miRNAs e os snoRNAs, os piRNAs também estão relacionados com a regulação gênica. Mais especificamente, eles atuam no silenciamento de genes capazes de se auto duplicar no interior do genoma.

Os ***small non-messenger* RNAs (snmRNAs)** [61] são classes de RNAs com funções de regulação.

O ***Small Cajal body-specific* RNA (scaRNA)** [29, 30] tem função similar à dos snoRNAs. Sua estrutura é formada por ambas as características dos tipos de snoRNAs: box C/D e box H/ACA.

2.2.2.2 ncRNAs longos (lncRNAs)

O lncRNA têm como características básica o tamanho, devem ser maiores do que 200 nucleotídeos, além do fato de que as proteínas não são sintetizados a partir deles [60, 115]. Então, lncRNAs às vezes têm pouca capacidade de codificação de proteínas [60, 93]. Os lncRNAs são transcritos que apresentam extremidades tanto de 5' para 3' como ao contrário, podendo sofrer *splicing*. Entretanto, não apresentam *Open Reading Frame* (ORF) suficiente para codificar proteínas, tendo seu tamanho variando de 200 a 100.000

pares de bases [101]. Recentemente, aumentou o volume de pesquisas visando a descoberta e caracterização de novos lncRNAs [46], os RNAs menos conhecidos até o momento. Não se sabe, por exemplo, se (e em qual proporção) esses lncRNAs podem ser processados e dar origem a novos RNAs regulatórios pequenos.

Apesar de serem menos conservados do que genes codificadores em relação à sequência de nucleotídeos, os lncRNAs apresentam uma conservação maior em suas estruturas secundárias [141]. Portanto, os lncRNAs vêm sendo recentemente considerados como reguladores chave de diversos processos biológicos [79, 99].

Os lncRNAs podem ser transcritos a partir de regiões distantes dos genes codificadores, dentro dos transcritos ou de genes a partir de *íntrons*. Eles podem exercer sua ação a partir da região de origem, regulando seus alvos [148].

Já os lncRNAs que são derivados da fita de DNA oposta à de um gene codificador são conhecidos como transcritos antisenso naturais (NATs) e regulam o gene ao qual o lncRNA se sobrepõe [101]. Assim, eles incluem os pseudogenes, que são cópias de um determinado gene que não vai produzir uma proteína [55].

A grande maioria dos lncRNAs que já possui sua função caracterizada são os envolvidos em regulação [28]. Esses lncRNAs são associados a um complexo de remodeladores de cromatina, ou seja, um grupo de genes que reestruturam os nucleossomos de modo a compactar mais ou menos a cromatina, determinando o nível de transcrição gênica de uma região definida do cromossomo. As interações entre proteínas e lncRNAs poderiam resultar em mudanças conformacionais que seriam úteis para distinguir a especificidade da região alvo [126]. De forma resumida, os lncRNAs serviriam como "guias" para os complexos remodeladores de cromatina, pois esses não possuem capacidade de ligação ao DNA, não reconhecendo suas regiões alvo de forma isolada [101].

Outras funções dos lncRNAs estão relacionadas ao processamento de mRNA (*splicing*) e ao aumento ou diminuição da capacidade de tradução de seus alvos [148].

A enumeração de todos esses ncRNAs seria muito extensa. A Tabela 2.3 traz uma lista de algumas classes de ncRNAs para exemplificar a diversidade de funções que eles desempenham.

Tabela 2.3: Alguns tipos de ncRNAs e suas funções conhecidas [36, 132]

Classes de ncRNAs		
Sigla	**Função**	**Referências**
tRNA	envolvidos na tradução de mRNAs	[112]
rRNA	RNA constituinte do ribossomo	[36]
snRNAs	envolvido no processo de *splicing*	[73, 138, 143]
snoRNAs	envolvidos na modificação do rRNA	[32]
miRNA	família putativa de genes reguladores da tradução	[92]
siRNA	moléculas ativas na interferência de RNA	[36]
piRNA	regulação de tradução e estabilidade de mRNA, entre outras	[16]
snmRNA	ncRNAs pequenos	[61]
scaRNA	tem função similar à dos snoRNAs	[29, 30]
lncRNA	regulam a expressão de genes, não têm potencial de codificação de proteínas	[60]

2.2.3 Ferramentas computacionais para anotação de ncRNAs

Nesta seção, são descritas algumas ferramentas computacionais para detecção de ncRNAs de classes conhecidas.

Os métodos computacionais para identificar ncRNAs sofrem de problemas similares aos dos métodos experimentais. A Bioinformática não possui métodos únicos para identificação e classificação de ncRNAs, embora alguns critérios sejam usados, como: o fato de que ncRNAs não possuem em geral ORFs longas; há, ao longo de sua sequência, ocorrências de códons de parada maiores do que a esperada [146]; provavelmente RNAs tem uma conservação em nível de estrutura secundária, e não da estrutura primária, o que invalida a detecção de ncRNAs por meio de ferramentas tradicionais usadas para caracterizar similaridade de DNA em proteínas [108, 120, 141]. Estudos que incorporam uso de códons e energia mínima de dobramento também são bem-sucedidos na identificação de ncRNAs [7, 141, 160].

Como os ncRNAs mais conhecidos possuem uma estrutura tridimensional complexa e têm funções tanto catalisadoras como estruturais [33], a atual tendência dos métodos de Bioinformática é recorrer a uma combinação de diversos métodos computacionais que caracterizem os ncRNAs por meio de diferentes métodos. Depois os biólogos, analisam todas as informações geradas pelos métodos para decidir quais RNAs provavelmente são não-codificadores [78, 96].

Sabendo que o problema na busca de ncRNA é essencialmente de classificação, a escolha de um método depende dos dados disponíveis das sequências que estão sendo analisadas [120, 147]. Uma desvantagem é a de que, para confirmar a predição feita *in silico* de um transcrito ser não-codificador, é necessária validação experimental no laboratório, e a observação de ausência de tradução não é conclusiva, já que o eventual transcrito poderia ser traduzido logo que exposto a outras condições, ambientais ou fisiológicas [78, 96].

Em seguida, descrevemos ferramentas para identificar e classificar ncRNAs.

BLAST

O programa *Basic Local Alignment Search Tool* (BLAST) [1] é bastante utilizado na comparação aproximada entre duas sequências. É um método de alinhamento local, que compara uma sequência com outras sequências com funções já definidas armazenadas em um banco de dados. O BLAST pode ser utilizado para identificar relação evolucionária, inferir funções ou identificar uma família de genes entre sequências.

O BLAST utiliza procedimentos heurísticos para produzir alinhamentos locais. Ele também calcula um valor esperado (*expect value* - $e - value$) que estima quantas identidades ocorreram com dado escore ao acaso, que ajuda o usuário a julgar a confiança no alinhamento produzido. O método do BLAST é dividido em três grandes etapas. Na primeira, são encontrados sequências pequenas de tamanhos fixados (*words*) que ocorrem na sequência de consulta. Na segunda etapa essas palavras são usadas para fazer busca pela mesma (*string*) em todas as sequências de um banco de dados (*subject*). Em seguida, são feitas extensões, com espaços (*gaps*), em ambos os lados da sequência de consulta em relação à sequência do banco de dados, mantendo um escore minimo. Essas extensões são então ligadas produzindo alinhamentos maiores, porém, ainda mantendo um escore mínimo. É importante ressaltar que o método BLAST usa matrizes de substituição (que indicam a probabilidade de mutação entre os aminoácidos ou bases ao longo

da evolução), para pontuar os pares de aminoácidos ou bases alinhados. As matrizes de substituição mais amplamente usadas são as *BLOcks of amino acid SUbstitution Matrix* (BLOSUM) [53].

O BLAST é constituído por uma série de programas:

- **blastp**: para comparação de sequências de aminoácidos com um banco de dados de proteínas;
- **blastn**: para comparação de sequências de nucleotídeos com um banco de dados de nucleotídeos;
- **blastx**: para comparação de sequências de nucleotídeos traduzidos em todas as ORFs, com um banco de dados de proteínas;
- **tblastn**: para comparação de sequências de proteínas com um banco de dados de sequências de nucleotídeos traduzidos em todas as suas ORFs;
- **tblastx**: para comparar as ORFs de sequências de nucleotídeos com todas as ORFs de um banco de dados de nucleotídeos.

Infernal

O *INFERence of RNA Alignment* (Infernal) é um método que utiliza uma abordagem baseada em Gramática Estocástica Livres de Contextos (SCFG, *Stochastic Context-Free Grammars*) [35, 123]. Essa ferramenta constrói perfis de RNA consenso chamados de modelos de covariância (CM, *Covariance Models*), que são casos especiais de SCFG, projetada para modelar sequências e estruturas de RNAs. O Infernal usa esses CMs para procurar as semelhanças entre as estruturas secundárias das famílias de RNAs do banco de dados Rfam [119] e a da sequência sendo investigada.

Cada CM é construído a partir do alinhamento múltiplo de sequências e de informações relacionadas à estrutura secundária consenso.

O Infernal compreende os programas *cmbuild*, *cmsearch* e *Rsearch*. A construção do CM, feita pelo *cmbuild*, requer como entrada um alinhamento múltiplo de RNAs no formato Estocolmo (*Stockholm*)[3] [34], gerando um arquivo de saída contendo o CM, o qual será usado por outras funções do Infernal. A busca em bases de dados por possíveis homólogos, feitas pelo *cmsearch*, requer duas entradas: um arquivo CM (obtido com o *cmbuild*), e um arquivo contendo as sequências a serem analisadas. O *cmsearch* busca as sequências que geraram *hits* com alta pontuação para o CM usado. É gerada uma saída contendo os alinhamentos para cada *hit* em um formato similar ao do BLAST [1]. Dentro da ferramenta Infernal, existe um módulo chamado *Rsearch*, que compara sequências de RNAs com um banco de sequências de RNAs. Desta forma, dada uma sequência de RNA, são feitas buscas em uma base de dados de nucleotídeos por RNAs homólogos. Esta busca é baseada tanto na estrutura primária quanto na estrutura secundária [68].

[3] É um formato de alinhamento múltiplo de sequências usado no Pfam (banco de dados com famílias de proteínas) e no Rfam (banco de dados com famílias de ncRNAs).

tRNAscan-SE

O programa tRNAscan-SE [83] é considerado um dos preditores de tRNAs mais precisos [74]. Ele combina três programas: dois preditores de tRNAs que buscam promotores de RNA polimerase III e características da estrutura secundária [39, 111], e um Modelo de Covariância [35] treinado com sequências de tRNAs. Os dois primeiros programas são rápidos e, quando combinados, possuem uma sensibilidade superior a aproximadamente 99%. Porém, tal combinação implica em uma taxa de aproximadamente 1.85 falsos positivos por MB, o que é aceitável para genomas pequenos, mas significa aproximadamente 5.500 falsos positivos no genoma humano.

O Modelo de Covariância é bastante sensível e específico, mas muito lento. Portanto, os dois primeiros preditores de tRNAs são utilizados (com baixa estringência) como filtros a fim de obter candidatos promissores de tRNAs de um genoma. Os candidatos são então analisados pelo Modelo de Covariância, altamente estringente. O resultado é um identificador de tRNAs apresentando alta sensibilidade (99-100%) e especificidade (com uma taxa de falsos positivos inferior a 0.00007 por Mb), com uma velocidade razoável (30 Kb/s).

SVM-Portrait

O SVM-Portrait [5, 6] é adequado para identificar ncRNAs de transcritomas incompletos ou de espécies cujas caracterizações não foram concluídas. Essa ferramenta utiliza métodos baseados em técnica de aprendizagem de máquina, particularmente Máquina de Vetor de Suporte (*Support Vector Machine* - SVM). O resultado do SVM-Portrait é a probabilidade de uma sequência não codificar uma proteína.

Vienna

O Vienna [56, 58] é um pacote de programas que geram ou comparam estruturas secundárias de RNAs. Esse pacote tem várias ferramentas, nas quais dobramentos são feitos utilizando um algoritmo de predição basendo na energia livre do RNA [161], e nas probabilidades de pareamento de bases [91].

O pacote RNAz realiza a predição de estrutura baseada na energia mínima livre (*Minimun Free Energy - (MFE)*) [58, 162]. É levado em consideração o fato de que as estruturas dos ncRNAs apresentam duas características: a estabilidade termodinâmica e a conservação da estrutura secundária [51]. Como primeiro critério, o RNAz calcula uma medida normalizada da estabilidade termodinâmica e a seguir uma pontuação (z-*score*) é gerada. Uma pontuação mais negativa indica que a sequência é mais estável do que a esperada ao acaso [51]. Como segundo critério, o RNAz prediz uma estrutura secundária consenso de um alinhamento usando o programa RNAalifold [58]. Mutações compensatórias (mutações que preservam um par de bases correto, por exemplo, substituição do par CG pelo par UA) são pontuadas, enquanto que mutações inconsistentes (a substituição do par CG por CA, por exemplo) adicionam penalidades. No final, é calculado o índice de conservação da estrutura (*structure conservation index - (SCI)*) [51]. Finalmente, o RNAz utiliza um algoritmo de aprendizado de máquina SVM, que foi treinado utilizando um vasto conjunto de ncRNAs conhecidos. Esta etapa utiliza os resultados do critério (z-*score*) para

classificar o alinhamento de entrada como "RNA estrutural" ou "outros" [51]. O RNAz computa a probabilidade de sequências serem ncRNAs.

O pacote RNAfold [82] foi projetado para fazer o dobramento bidimensional de uma sequência de RNA, baseado em MFE e usando o algoritmo de programação dinâmica proposto originalmente por Zuker e Stiegler [162]. Além do dobramento MFE, probabilidades de equilíbrio de pareamento de bases são computados por método McCaskill [91], pelo algoritmo de função de partição. O RNAfold oferece várias possibilidades de controlar a estrutura espacial por parte do usuário, como locais da estrutura secundária onde ocorre o pareamento de nucleotídeos para a formação das hélices.

Por ultimo, o pacote RNAalifold [58] constrói uma estrutura bidimensional consenso, a partir do alinhamento múltiplo de sequências de RNA. O algoritmo utiliza informações termodinâmicas e filogenéticas para determinar a estrutura da predição. Uma estrutura secundária de consenso é inferida a partir do alinhamento.

RNAmmer

O RNAmmer [71] é uma ferramenta de predição de rRNAs que utiliza os bancos de dados 5S *ribosomal database* e *European ribosomal RNA database* com base em Cadeias de Markov.

A Tabela 2.4 mostra os programas para predição de ncRNAs descritas nesta seção.

Tabela 2.4: Ferramentas computacionais

Nome	Descrição	Referências
BLAST	Realiza alinhamento local de sequências primárias	[1]
Infernal	Baseado em Gramática Estocástica Livres de Contextos e Modelo de Covariância	[35]
tRNAscan-SE	Usa modelo de covariância na predição de tRNAs	[83]
SVM-Portrait	Identifica ncRNAs de transcritomas incompletos	[5]
Vienna	Compara estruturas secundárias de RNAs	[58]
RNAmmer	Usa cadeias de Markov na predição de rRNAs	[71]

2.2.4 Bancos de dados

Nesta seção descrevemos os bancos de dados que contém ncRNAs. Estes repositórios são criados tanto a partir de dados experimentais quanto de dados computacionais [137].

NONCODE

Todos os ncRNAs do NONCODE [105] foram filtrados automaticamente da literatura e do GenBank [45] e, em seguida, tratados manualmente. O NONCODE inclui quase todos os tipos de ncRNAs, exceto tRNAs e rRNAs. Mais de 80% das entradas do NONCODE estão baseadas em dados experimentais. A primeira versão do NONCODE (v1.0) continha 5.339 sequências de 861 organismos, hoje a versão v3.0 contém mais de 411.552 [77].

RNAdb

O RNAdb [121] é um banco de dados de ncRNAs de mamíferos que contém sequências e anotações de milhares de ncRNAs, mas a maioria com papéis ainda não conhecidos [109].

miRBase

O miRBase [98] é um banco de dados de miRNAs [50].

snoRNA Database

O snoRNA Database [76, 136] contém snoRNAs humanos do tipo H/ACA *box* e C/D *box*.

Plant snoRNAs Database

O Plant snoRNA Database contém snoRNAs de plantas [19, 135].

fRNAdb

O fRNAdb [100] integra um conjunto de outras base de dados, dentre outros o NONCODE e o RNAdb.

Rfam

O Rfam é uma base de dados curada (revisada e supervisionada), que contém informações sobre milhares de famílias de ncRNAs. Esta base de dados contém duas classes distintas de dados: os perfis de modelos de covariância (CMs) e os alinhamentos semente (*seed alignments*). Como dito anteriormente, os CMs são modelos estatísticos resultantes da combinação de informações tais como estrutura secundária e sequências primárias, representadas pelo alinhamento múltiplo de sequências. Cada perfil de CM corresponde a uma família de ncRNAs. Já os alinhamentos semente estão contidos dentro de um arquivo no formato Estocolmo e contém os membros representativos de cada família de ncRNAs gerados através de diversos alinhamentos estruturais [22, 49]. Os arquivos Rfam podem ser obtidos do site de FTP do Sanger [119]. Nesse trabalho foi utilizado o Rfam 12.0 de Julho de 2014 com 2.450 famílias.

A Tabela 2.5 mostra os bancos de dados descritos nesta seção.

Tabela 2.5: Bancos de Dados de ncRNAs

Nome	Descrição	Referências
NONCODE	Contém quase todos os tipos de ncRNAs, exceto tRNAs e rRNAs	[105]
RNAdb	Contém ncRNAs de mamíferos	[121]
mirBASE	Contém miRNAs	[98]
snoRNA	Contém snoRNAs humanos do tipo H/ACA box e C/D *box*	[136]
snoRNAs de Plantas	Contém snoRNAs de plantas	[135]
fRNAdb	Contém vários bancos de dados	[100]
Rfam	Contém milhares de famílias de ncRNAs	[22, 119]

2.3 Características e desafios na anotação de ncRNAs

Nesta seção, discutimos aspectos importantes para a anotação *in silico* de ncRNAs.

Atualmente, a anotação de ncRNAs envolve três principais tipos de métodos computacionais: predição de estrutura secundária [48, 57, 63], comparação de estrutura secundária [36, 58], identificação e classificação de RNAs não-codificadores [48].

Como dito anteriormente, um ncRNA normalmente requer uma estrutura tridimensional específica para desempenhar sua função [129, 142]. Uma vez que a estrutura tridimensional é determinada pela estrutura secundária, a última é utilizada como aproximação no estudo da relação estrutura-função. A estrutura secundária, por sua vez, é definida pela sequência primária. Portanto, ferramentas para predizer a estrutura secundária a partir de uma sequência primária de RNA são úteis para estudar sua função. Quando um conjunto de RNAs homólogos é conhecido, sua estrutura secundária consenso pode ser predita com maior confiabilidade. Além disso, a conservação de domínios estruturais em diferentes espécies constitui evidência adicional de que esses domínios estão relacionados à função específica desta sequência.

A comparação de estruturas secundárias pode servir a muitos propósitos. Por exemplo, ela pode ser utilizada para classificar um RNA como membro de uma família, comparando sua estrutura com a estrutura consenso das várias famílias conhecidas. Além disso, se a função de um único RNA (ou de uma família) não é conhecida, ela pode ser inferida comparando a estrutura desse RNA (ou consenso, no caso de uma família) com um banco de dados com estruturas já anotadas funcionalmente. A comparação de estruturas pode também ser utilizada para detectar a ocorrência de diferentes estruturas estáveis de uma mesma molécula (o que pode indicar a presença de alterações conformacionais possivelmente relacionadas com a função do RNA), e portanto pode ser usada para predizer mutações em uma sequência de RNA que causam rearranjos na estrutura secundária ou, ainda, para comparar um conjunto de estruturas para escolher um representante.

Finalmente, predição e comparação de estruturas podem ser utilizadas para buscar ncRNAs em genomas, tanto através de buscas de RNAs homólogos a um candidato específico ou de ncRNAs em geral, incluindo novas famílias ainda desconhecidas.

Capítulo 3

Sistema Multiagente

Neste capítulo, é feita uma breve apresentação dos conceitos relacionados a área de Sistema Multiagente. Na Seção 3.1 será apresentado o conceito de agentes inteligentes e suas características. Na Seção 3.2, será detalhada a área de Sistema Multiagente. Por fim, na Seção 3.3, são apresentados alguns motores de inferência para o raciocínio dedutivo de agentes inteligentes.

3.1 Agentes Inteligentes

Embora não exista uma única definição para agente de software, existe a noção de que a autonomia é essencial em um agente inteligente. Baseado nesta noção de autonomia [122, 157], foi adotada a seguinte definição de agente neste trabalho: Um agente é uma entidade capaz de perceber o ambiente onde está inserido, através de sensores e de agir sobre esse ambiente através, de atuadores (Figura 3.1).

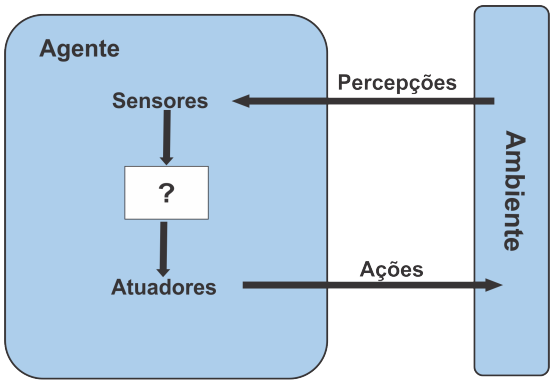


Figura 3.1: Arquitetura de um agente inteligente (adaptado de [122]).

A definição apresentada mostra que um agente é uma entidade capaz de perceber a situação corrente do ambiente. Este ambiente pode ser real ou simulado computacionalmente [122]. Note que cada agente exibe duas características fundamentais: (i) ele é capaz de agir de forma autônoma, tomando decisões que levam à satisfação dos seus objetivos; e (ii) é capaz de interagir com outros agentes utilizando protocolos de interação social inspirados nos humanos, incluindo pelo menos algumas das seguintes funcionalidades - coordenação, cooperação, competição e negociação [122].

Por ser uma entidade autônoma, um agente percebe o seu ambiente e toma decisões baseado em suas próprias experiências, adquiridas na interação com o ambiente ou com outros agentes. Portanto, um agente deve buscar melhorar seu desempenho de predição de ncRNA com base nas suas percepções e no conhecimento armazenado. A autonomia e flexibilidade do comportamento do agente pode ser alcançada com base no processo de resolução de problemas, tomada de decisões e aprendizado. Desta forma, a melhor racionalidade é atingida quando o agente consegue melhorar um resultado, seja com base em um modelo probabilístico, por exemplo baseado em incertezas, onde o agente escolhe o melhor resultado, ou aprende a partir de experiências adquiridas [64].

Conforme exposto, o agente inteligente está inteiramente ligado a suas percepções, ações, objetivos e ao ambiente em que está inserido [157]. Um agente inteligente com raciocínio dedutivo tem que apresentar características que permitam avaliar sua situação e tomar decisões a partir de um conjunto de regras de inferência [12]. Esse agente pode executar vários comportamentos apropriados para o modelo de mundo que o sistema representa. Além disso, é capaz de interagir com outros agentes e evoluir, alcançando novos comportamentos. Segundo [157], as principais características de um agente inteligente são:

- autonomia: pode trabalhar independentemente no ambiente em que está inserido, consegue aprender por experiência e alterar seu comportamento;
- reatividade: tem a capacidade de reagir ao longo do tempo com base em suas experiências e percepções, respondendo às mudanças que ocorrem no ambiente;
- proatividade: tem a capacidade de se comportar direcionado ao alcance de objetivos, adotando iniciativas para satisfazer os objetivos definidos; e
- sociabilidade: consegue interagir com outros ambientes e agentes, utilizando protocolos de comunicação e interação.

Dependendo do ambiente em que o agente está inserido, as características do ambiente podem exigir um pouco mais de racionalidade dos agentes. De acordo com [122], os tipos básicos de agentes que incorporam os princípios subjacentes dos sistemas baseados em agentes inteligentes são:

- reativo simples: percebe o ambiente em que está inserido e responde de forma hábil às mudanças que ocorrem nele. Esse tipo de agente é o mais simples devido ao seu comportamento reativo, pois simplesmente escolhe uma ação com base em suas percepções (Figura 3.2);
- reativo baseado em modelo: pode lidar com ambientes parcialmente observáveis. Deve manter um estado interno que depende do seu histórico de percepções e consiga refletir os aspectos não observados pelo estado atual; esse tipo de agente usa um modelo de mundo. Para atualizar as informações no decorrer do tempo o agente requer dois tipos de conhecimento: (i) informações de como o mundo evolui; (ii) algumas informações sobre como as ações do agente afetam o mundo. Esse conhecimento do mundo é chamado de modelo do mundo. A Figura 3.3 mostra a estrutura de um agente reativos baseado em modelo com seus estados internos, mostrando como a percepção vigente é combinada com um estado interno antigo para gerar a descrição do estado atual do mundo;

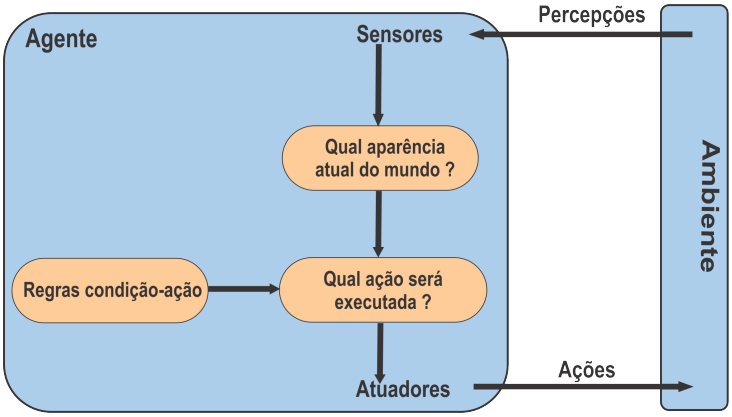


Figura 3.2: Agente reativo simples (adaptado de [122]).

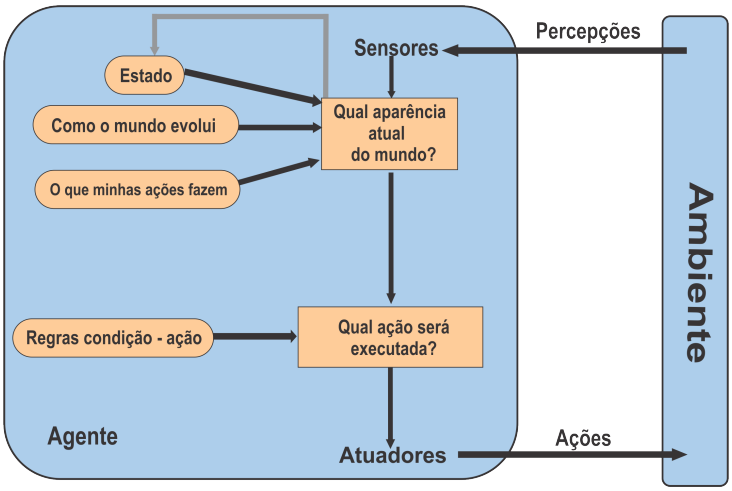


Figura 3.3: Agente reativo baseado em modelo (adaptado de [122]).

- baseado em objetivo: trabalha com um conjunto de objetivos alcançáveis a partir de um registro de estado interno. Suas ações são baseadas em objetivos levando-se em consideração a análise do estado do ambiente no momento, desta forma o agente precisa de algum tipo de informação de objetivo que descreva a situação que deseja alcançar. Entretanto, o objetivo não garante o melhor comportamento para o agente, apenas a distinção entre estados objetivos e não objetivos garante a sua eficiência. Embora pareça menos eficiente, ele é mais flexível, pois o conhecimento que dá suporte as suas decisões é representado explicitamente e pode ser modificado (Figura 3.4);
- baseado em utilidade: busca definir um grau de satisfação com os estados atuais do mundo. Verifica quanto o agente está "feliz" com aquele estado. Esse grau de satisfação é mensurado por uma função de utilidade, a qual vai atribuir um grau de satisfação ao agente no momento em que se encontra. Uma especificação de função de utilidade permite decisões racionais de dois tipos de casos, onde os objetivos são: (i) conflitantes, somente um deles pode ser alcançado, a função de utilidade especifica a troca apropriada; (ii), quando existem muitos objetivos que o agente pode escolher, mas nenhum dos objetivos pode ser alcançado com certeza. Neste

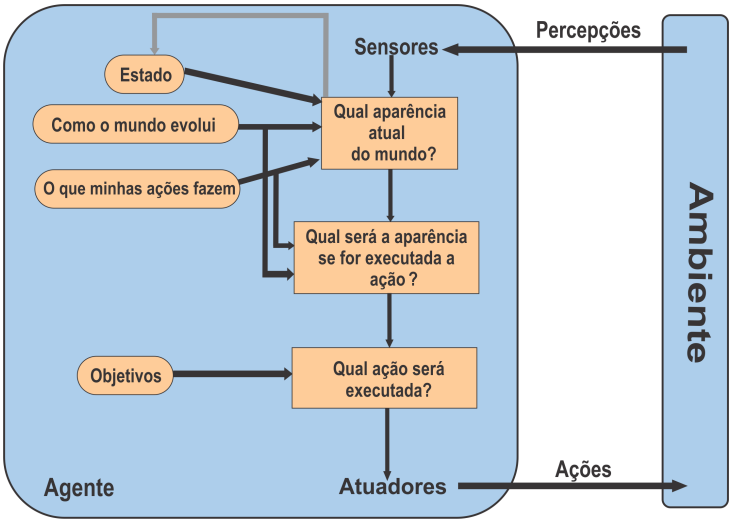


Figura 3.4: Agente baseado em objetivo (adaptado de [122]).

caso, a utilidade provê uma forma em que a probabilidade de sucesso pode ser ponderada considerado a importância dos objetivos.

Conhecer o estado atual não é sempre suficiente para decidir o que fazer. Objetivos sozinhos não são realmente suficientes para gerar comportamento de alta qualidade na maioria dos ambientes. Objetivos apenas provêem uma distinção binária crua entre os estados "feliz" e "infeliz", enquanto uma medida de performance mais geral pode permitir uma comparação de diferentes estados do mundo, de acordo com quão feliz eles podem fazer o agente e se eles podem ser alcançados. A Figura 3.5 apresenta a estrutura do agente baseado em utilidade; e

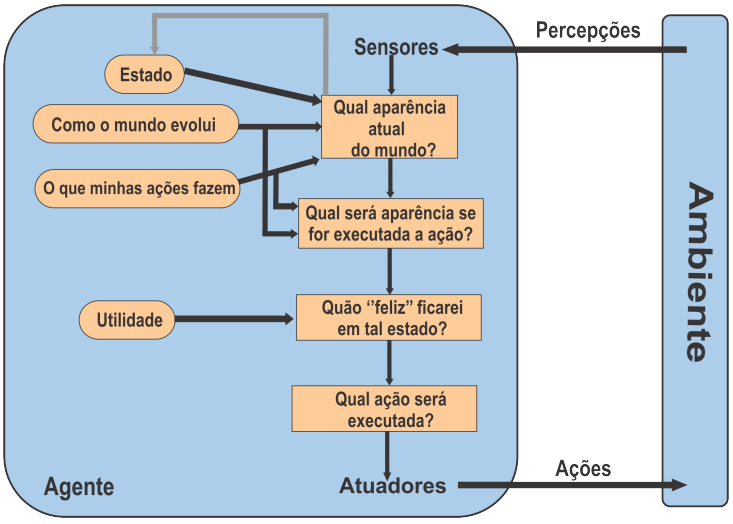


Figura 3.5: Agente baseado em utilidade (adaptado de [122]).

- baseado em aprendizado: o agente mantém suas percepções passadas e as que poderão ser usadas em sua próxima ação. Podem atuar em ambientes totalmente desconhecidos e se tornar mais eficiente do que o seu conhecimento inicial poderia

permitir. Uma arquitetura híbrida pode ser utilizada na maioria dos problemas, onde e feita uma combinação de aspectos deliberativos e reativos. A Figura 3.6 apresenta a estrutura deste tipo de agente. O componente gerador de problemas é responsável pela geração de novas soluções de problemas baseados no aprendizado realizado e nos objetivos de aprendizagem definidos. O componente elemento de aprendizagem é responsável pela melhora, enquanto o desempenho de predição de ncRNA é responsável pela seleção de ações externas. O elemento de aprendizagem é vital nesta arquitetura, pois utiliza *feedback* crítico de como o agente está atuando e determinando como o elemento de desempenho deve ser modificado para atuar melhor no futuro.

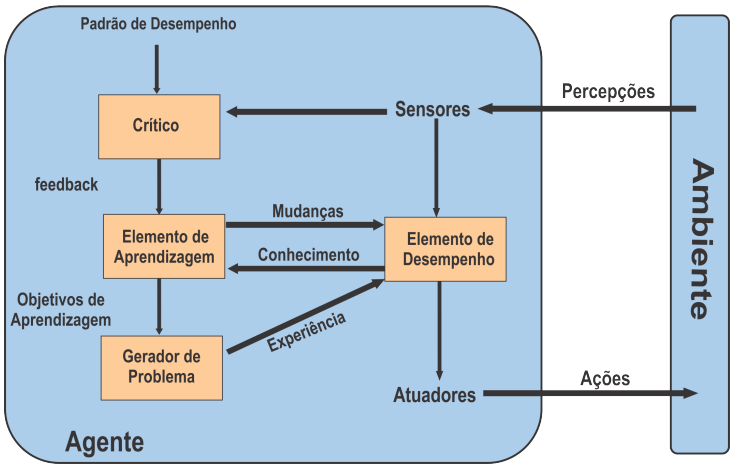


Figura 3.6: Agente baseado em aprendizado (adaptado de [122]).

3.2 Fundamentos

Um Sistema Multiagente (SMA) inclui diversos agentes que interagem ou trabalham em conjunto, podendo compreender agentes de tipo homogêneo ou heterogêneo. Cada agente opera assincronamente com respeito aos outros agentes [157]. Para que um agente possa operar como parte do sistema, faz-se necessária a existência de uma infraestrutura que permita a comunicação e interação entre os agentes que compõem o SMA, podendo ser feita de forma direta (comunicação explícita) ou de modo indireto (emissão de sinais através do ambiente). Uma organização define todas as restrições aplicadas aos agentes pertencentes a uma determinada sociedade, ou seja, os meios que o projetista do sistema pode garantir que cada agente alcançará a resolução dos problemas propostos [113].

Um SMA baseia-se na ideia que um ambiente de trabalho ser cooperativo incluindo componentes de software, os quais tratam com problemas que seriam difíceis de serem resolvidos usando uma abordagem de computação tradicional centralizada, apresentando diferentes esferas de influência ligadas por relações organizacionais [157]. A Figura 3.7 mostra que adotar uma estratégia orientada a agentes viabiliza a decomposição do problema em componentes múltiplos e autônomos, os quais podem interagir de forma flexível para alcançar seus objetivos, conforme interação entre organizações de agentes com diferentes esferas de influência no ambiente.

Existem alguns contextos organizacionais que dão suporte à interação entre os agentes. Eles podem ser *peers* trabalhando juntos em equipe, ou pode haver um agente gerenciando outros agentes. Os modelos de abstração que definem o conjunto da orientação a agentes são: agentes, interações e organizações [65].

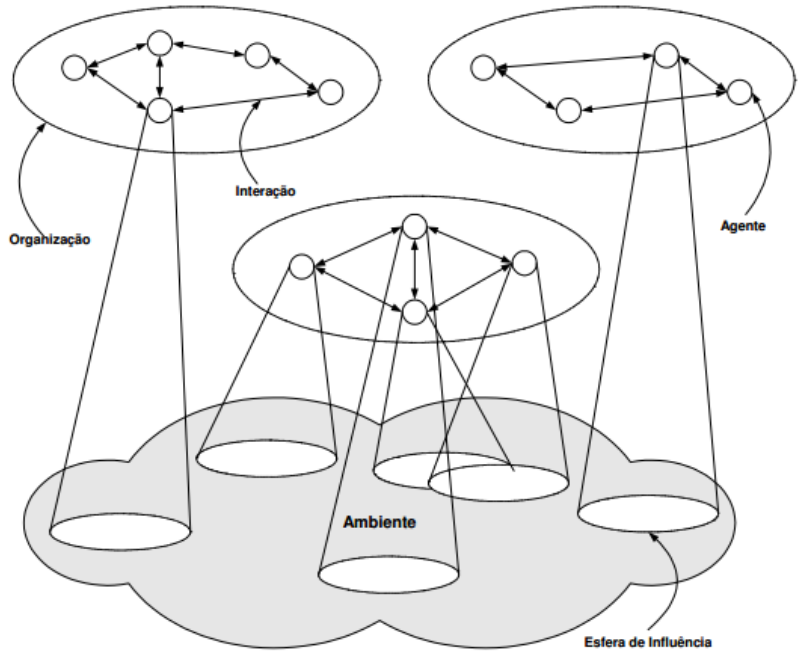


Figura 3.7: Estrutura típica de um SMA (adaptado de [157]).

Pode-se definir um sistema como sendo multiagente quando ele possui determinadas características: um ambiente, um conjunto de agentes, um conjunto de objetos, as interações entre os próprios agentes e um conjunto de operações que podem ser realizadas.

Assim, dado um determinado SMA, denomina-se agente cada uma de suas entidades ativas no sistema. O conjunto formado por esses agentes é chamado de sociedade. O termo ambiente representa as entidades passivas dentro do sistema. Um agente recebe informações e raciocina sobre o ambiente, sobre outros agentes e decide quais ações deve realizar e quais objetivos deve seguir. Um agente é uma entidade ativa, ou seja, capaz de controlar suas ações, diferentemente das noções que são consideradas estáticas, tais como módulos, conjunto de regras e bases de conhecimentos.

Os sistemas multiagentes costumam ser caracterizados didaticamente em duas classes, conforme apresentado na Seções 3.2.1 e 3.2.2. A primeira é denominada SMA Reativo, uma classe que trabalha com o desenvolvimento de sistemas que utilizam um grande número de agentes simples, os quais buscam a resolução de um determinado problema. Já a segunda abordagem, denominada SMA Cognitivo, trabalha com poucos agentes buscando resolver tarefas mais complexas que os reativos. No entanto, podem existir sistemas híbridos, onde coexistem os dois tipos de agentes [38].

3.2.1 Sistema Multiagente Reativo

O SMA reativo é um sistema baseado em modelo de organização biológica ou etológica tais como: formigas, cupins, abelhas, entre outros. O SMA reativo funciona por par-a-par, estímulo-resposta, ou seja, ação-reação. As principais características deste tipo de agente e dos sistemas são:

- o conhecimento não é representado de forma explícita: o agente tem seu conhecimento implícito em regras de comportamento e sua manifestação se externa através do seu comportamento em interação com os outros agentes;
- o ambiente não é representado internamente: o comportamento de cada agente tem como base o que ele percebe ao receber estímulos do ambiente. O ambiente não tem representação interna explícita para o agente;
- não existe um registro/memória das ações: não mantém um histórico das ações, assim, o resultado de uma determinada ação passada não vai influenciar diretamente na decisão de uma ação que vá ocorrer no futuro;
- organização etológica: é similar à observada por animais que vivem em grandes comunidades; e
- muitos membros: em geral, possuem um grande número de agentes, com populações que podem chegar à ordem de milhares de membros.

Resumidamente, os agentes reativos são muito simples e não possuem representação do ambiente, sendo que as reações dependem unicamente da percepção do ambiente. O SMA reativo concebe o problema como sendo um conjunto de agentes interagindo/comunicando entre si, onde cada um destes possui seus objetivos individuais. Uma forma usual de representar os comportamentos dos agentes é por meio de um conjunto de regras de produção.

3.2.2 Sistema Multiagente Cognitivo

O SMA cognitivo é baseado em modelos de organização social de sociedades humanas, tais como: grupos, hierarquias, mercados, entre outros. Esse tipo de agente possui uma representação explícita do ambiente e dos membros que estão na comunidade e podem raciocinar sobre as ações que foram tomadas no passado e assim planejar as futuras ações. Os agentes cognitivos podem ainda interagir com os demais membros da sua comunidade através de linguagens e protocolos de comunicação, podem utilizar estratégias sofisticadas de negociação. As principais características associadas ao SMA cognitivos são:

- o ambiente é representado de forma explícita e os agentes que podem interagir com outros agentes em sociedade;
- podem manter um histórico com as interações e ações passadas, graças a uma memória, sendo capazes de planejar suas ações futuras;
- usa a percepção para examinar o ambiente, e o mecanismo de comunicação que permite a troca de mensagens entre os agentes. A comunicação entre os agentes pode ser feita de modo direto, através de envio e recebimento de mensagens;

- possui um mecanismo de controle deliberativo. Os agentes cognitivos raciocinam e decidem em conjunto sobre quais ações executar, quais planos seguir e que objetivos devem alcançar;
- em geral, modelos de organizações de SMA cognitivos usam modelos sociológicos como, por exemplo, as organizações humanas; e
- usualmente um SMA cognitivo contém poucos agentes, na ordem de algumas dezenas no máximo.

Normalmente esse tipo de agente apresenta uma complexidade computacional elevada, sendo caracterizado por apresentar um comportamento inteligente tanto em comunidade quanto isoladamente. Geralmente as comunidades são compostas por um número pequeno de participantes. Esse tipo de agente possui um alto grau de determinação, pode decidir por motivação própria, levando em consideração quando e sob que condições ele deve escolher uma ação. Na maioria dos casos esses agentes precisam interagir com outros agentes para atingir seus objetivos, por não possuírem habilidades ou recursos suficientes para solucionar o problema sozinho, ou por terem interdependência em relação aos outros agentes.

As interações entre os agentes têm como objetivo fazer outros agentes tomarem suas decisões conscientes em relação ao que podem causar aos demais agentes envolvidos. Como esse tipo de agente não possui um controle direto sobre os outros, faz-se necessário utilizar uma estratégia de cooperação para aglutinar agentes na realização de uma dada tarefa, formando assim um SMA para solução de problemas através de ação cooperativa [102].

De forma resumida, podemos dizer que os agentes cognitivos, diferentemente dos reativos, possuem de forma detalhada e explícita uma representação do ambiente, além de possuir um histórico para auxiliar na tomada de decisão. Já os agentes reativos trabalham de forma diferente, eles possuem uma comunicação direta e procuram fazer um controle não deliberativo, sendo assim necessário uma quantidade considerável de agentes no sistema para produzir, um comportamento complexo.

3.2.3 Caracterização do Ambiente

Em [152] encontramos a definição de ambiente como uma abstração de primeira classe, a qual fornece as condições para os agentes existirem, além de controlar o acesso entre eles e o acesso aos recursos. Os ambientes provêm informações, as quais são captadas pelas percepções dos agentes. Segundo [122], os ambientes podem ser classificados quanto a suas propriedades:

- acessível ou inacessível: um ambiente acessível é aquele no qual o agente pode obter informações completas, precisas e atualizadas sobre o seu estado. Caso contrário, é dito inacessível. Quanto mais acessível um ambiente for, mais simples será o projeto e a construção do agente, mas a maioria dos ambientes de mundo real não são totalmente acessíveis;
- determinístico ou não-determinístico: um ambiente determinístico é aquele no qual uma ação tem um único efeito possível. Quando não é possível saber o estado que o ambiente assumirá após a execução de uma ação, dizemos que o ambiente é

não-determinístico. O não-determinístico pode encerrar incertezas sobre os estados resultantes da execução de ações;

- estático ou dinâmico: um ambiente é estático para um agente quando permanece inalterado até o momento em que ele executa alguma ação. Em um ambiente dinâmico, além das ações desempenhadas pelos agentes, existem processos que operam sobre o ambiente, alterando o estado do mesmo de forma dinâmica; e
- discreto ou contínuo: um ambiente discreto é aquele que possui um número fixo ou finito de estados a serem percebidos e correspondentes ações. Um ambiente contínuo é aquele no qual é possível assumir um número não fixado de estados.

Com base nas propriedades dos ambientes pode-se concluir que os ambientes parcialmente acessíveis, não-determinísticos, dinâmicos e contínuos são os mais complexos para projetar e implementar agentes inteligentes.

3.2.4 Protocolo de Comunicação

Na definição apresentada na Seção 3.1, assume-se que um agente inteligente possui a habilidade social para interagir/comunicar com outros agentes que estão no ambiente em que se insere. Sendo assim, uma importante característica do SMA é a comunicação. Conforme [122], comunicação é a troca de informação feita pelos agentes de forma intencional com percepção de sinais extraídos de um ambiente compartilhado. Na Figura 3.8 é representado o esquema genérico de um agente com capacidade de comunicação.

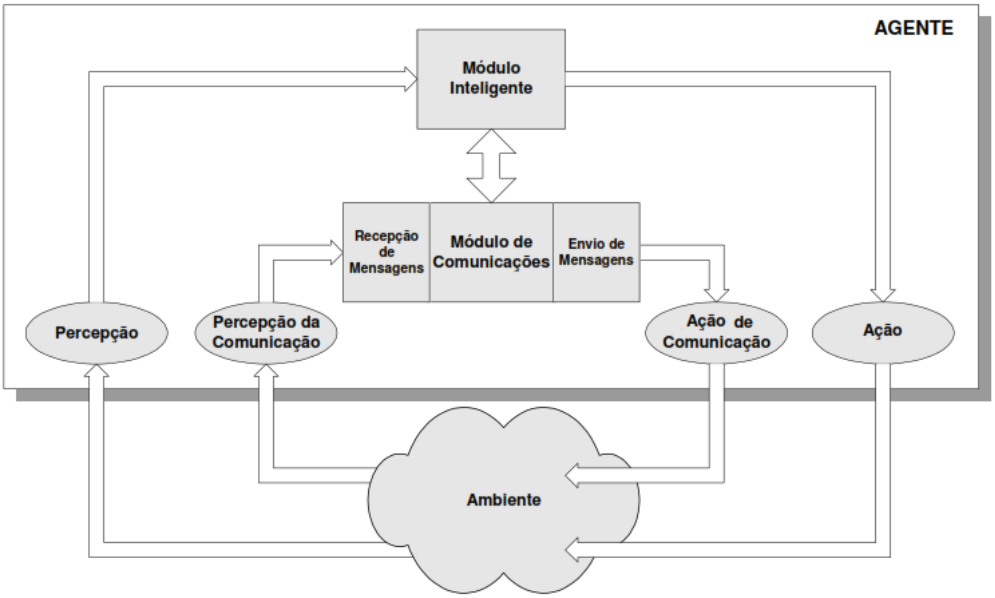


Figura 3.8: Agente com capacidade de comunicação (adaptado de [18]).

A comunicação tem dois fins principais: (i) compartilhamento do conhecimento, informações, crenças com os outros agentes; e (ii) coordenação das atividades entre os agentes. Então, a realização de comunicação que permita atingir essas duas metas, requer a definição de uma linguagem comum ou compartilhada com os agentes que estão no ambiente. Uma linguagem de comunicação apresenta as seguintes características:

- sintaxe: é a relação existente entre as palavras dentro de uma unidade, usando assim, parte da gramática que contém as regras que são relativas à um conjunto/combinação de palavras em unidades maiores;
- semântica: é a combinação feita com símbolos e seus respectivos significados, buscando fazer um estudo sobre o significado para as palavras e dos enunciados;
- vocabulário: explicação sucinta com definições de uma lista de vocábulos de linguagem, que em geral estão desacompanhadas de sua respectiva definição;
- pragmática: é o conjunto de regras de ações que define para que a interpretação dos símbolos através da comunicação é feita; e
- modelo de domínio de discurso: é o significado que um conjunto de símbolos pode assumir depois da interpretação feita dentro de um contexto específico.

Em 1962, o filósofo John Austin [31] desenvolveu a teoria *Speech Acts* (atos de fala), cujo trabalho foi estendido por John Searl em 1969 [130], que recebeu o nome *Speech Acts II*. Estes estudos dos atos de fala tem servido de inspiração em estudos relativos à criação de linguagens e protocolos de comunicação entre agentes.

De acordo com [31] os atos de fala podem ser distribuídos em três aspectos:

- ato locucionário: é uma declaração relacionando a um meio físico (por exemplo o ato fonético, ou a escrita);
- ato ilocucionário: é a intenção ou compreensão associada à declaração do locutor; e
- ato perlocucionário: é a ação resultante ou efeito esperado com a locução.

Esses atos podem ser caracterizados como enunciados que produzem efeitos nos sentimentos, pensamentos ou ações dos ouvintes, e ainda que podem ser realizados com a intenção de produzir tais reações. Atos desse tipo podem ser, por exemplo: respostas, perguntas, requisições, declarações, ou seja, atos que não podem ser diferenciados como o simples verdadeiro e falso, ou que deixam dúvidas quando ao seu propósito. A esse atos ou sentenças Austin chamou de performativas de comunicação.

Segundo [90], no começo de 1990, várias linguagens foram desenvolvidas pelo governo americano buscando melhorar a comunicação entre agentes, financiados pelo Defense *Advanced Research Project Agency* (DARPA), através da operação *Knowledge Sharing Effort* (KSE). A missão do grupo era desenvolver um protocolo através do qual fosse possível fazer a troca e representação do conhecimento entre sistemas de informação autônomos. Então, foi gerada uma primeira linguagem de comunicação pela KSE, que tinha uma grande abrangência em sua utilização, tendo sido denominada de *Knowledge Query and Manipulation Language* (KQML).

O KQML é uma linguagem de comunicação para agentes que possui um protocolo para a troca de informações e conhecimento, tomando como base as mensagens. Essa linguagem define um formato para as mensagens, não se preocupando com o conteúdo a ser enviado pelos agentes. Uma mensagem KQML pode ser definida como uma ação performativa[1] e um conjunto de parâmetros; sendo seus enunciados não classificados como

[1] É o ato que representa a vontade do agente sobre a informação contida na mensagem

verdadeiros ou falsos, como na lógica proposicional. Em [130] é feita uma classificação em cinco classes de performativas da linguagem:

- representativas: é feita uma comunicação entre locutor e receptor, onde é expressada uma crença do receptor e o locutor acredita na verdade do enunciado;
- comissivas: expressa o comprometimento do locutor com uma ação futura;
- declarativas: faz a mudança do estado do ambiente e depois afirma os fatos;
- diretivas: faz uma solicitação, que expressa seu pedido ou um comando específico;
- expressivas: busca fazer um agradecimento, em que pode mostrar seu estado psicológico expressando assim suas desculpas; e
- vereditas: expressam conclusão ou julgamento.

Atualmente, a linguagem de comunicação mais utilizada para a comunicação entre agentes foi padronizada pela *Foundation for Intelligent Physical Agents* (FIPA), sendo denominada *Agent Communication Language* (ACL) [41], a qual incorporou alguns aspectos da linguagem KQML e tem sua origem no *speech acts*. No ano de 1996, a FIPA[2] começou o desenvolvimento de padrões para sistemas baseados em agentes. O principal objetivo da FIPA com adoção do ACL era o desenvolvimento de uma linguagem padrão de comunicação de agentes. Sendo similar ao KQML, a ACL foi definida como linguagem para a troca de mensagens, possuindo com 20 performativas para as interpretações das mensagens, não tendo restrições ou indicações de linguagem para o conteúdo das mensagens. Então, como característica primária da FIPA ACL propôs a possibilidade de utilização de gerentes de conversação por meio de protocolos de interação pré-definidos. Mesmo com semelhanças, a mais importante diferença entre KQML e FIPA ACL é o conjunto de performativas que cada linguagem provê.

A FIPA propôs também uma semântica formal para a comunicação entre os agentes. Essa semântica foi criada em relação a uma linguagem formal denominada *Semantic Language* (SL). Com essa linguagem os agentes conseguem representar suas crenças, seus desejos, intenções e ações executadas, buscando mapear cada mensagem ACL a uma fórmula de SL, definindo assim uma restrição que o remetente da mensagem deve satisfazer se quiser ser considerado em conformidade com o padrão FIPA ACL. Essa restrição é conhecida como *condição de viabilidade*.

A Figura 3.9 apresenta um breve contexto sobre a organização das especificações FIPA, a saber:

- comunicação entre agentes: lidam com a estrutura das mensagens utilizadas, protocolos que fazem a interação entre agentes e a representação do conteúdo das mensagens;
- gerência de agentes: lidam com o controle e gerência de agentes em uma plataforma ou entre diferentes plataformas de agentes; e

[2]FIPA é uma organização filiada ao IEEE *Computer Society*, cujo objetivo é promover e padronizar a tecnologia orientada a agentes [4]. http://www.fipa.org/about/index.html

- transporte de mensagens: lidam com o transporte e representação de mensagens utilizando protocolos de rede diferentes, incluindo ambientes.

Neste trabalho, a comunicação dos agentes utiliza os padrões e linguagem FIPA ACL. Esse padrão é formado por uma coleção de especificações que promove a comunicação entre agentes heterogêneos em suas atividades. Essa especificação é dividida por um conjunto completo com as categorias apresentadas na Figura 3.9. A comunicação entre os agentes é a categoria mais importante deste modelo de arquitetura de SMA.

Figura 3.9: Organização das especificações FIPA [41].

De acordo a FIPA, agente é um processo computacional que implementa a funcionalidade comunicativa autônoma de uma aplicação. Um agente deve possuir um proprietário, uma organização ou um usuário humano, e tem que ter um conceito de identidade chamado *Agent Identifier* (AID) que o rotula, permitindo que o mesmo possa diferenciar-se dos demais agentes, além de permitir a definição de remetente e destinatários durante a troca de mensagens entre eles.

Como já descrito, pode-se observar que os agentes precisam de uma linguagem de comunicação, que permita trocar mensagens com base em um vocabulário comum e bem definido para entender em qual contexto estão trabalhando.

3.2.5 Protocolo de Interação

Os protocolos de interação são usados para especificação do comportamento entre os agentes durante a interação. Com o intuito de garantir uma forma efetiva de operação, possibilitando uma interação produtiva, faz-se necessário a definição de um protocolo de comunicação e interação. Por exemplo, um protocolo de interação pode especificar que, quando um agente recebe uma mensagem *Request* com um dado pedido, só pode responder com a mensagem *Not-Understood* (se não percebe algum aspecto da mensagem recebida), ou com a mensagem *Refuse* (se não aceita o pedido que lhe é feito), ou com a mensagem *Agree* (se aceita o pedido que lhe é feito).

A utilização de protocolos de interação auxiliam no projeto de aplicações baseadas em agentes porque permite reduzir o número de alternativas que um agente deve considerar em cada passo da sua interação com outros. Um importante protocolo usado no ambiente de negociação é o protocolo *Contract net* [134]. O *Contract net* usa uma estrutura descentralizada de negociação, onde seus agentes podem demandar serviços, recursos ou informações para outros agentes.

Com base nisso, percebe-se que a ideia principal é a de que quando um agente não consegue resolver um problema sozinho usando o seu conhecimento ou recursos locais, então ele vai decompor problemas em subproblemas e assim, tentar encontrar agentes dispostos a auxiliar e com o conhecimento necessário para resolver cada um dos subproblemas. Essa abordagem é conhecida como dividir-para-conquistar.

A definição do protocolo estabelece quando certas tarefas podem ser desempenhadas simultaneamente, por mais um único agente, enquanto outras tarefas só podem ser desempenhadas por um agente (em cada interação). Enfim, a definição do protocolo de interação, juntamente com o protocolo de comunicação, auxiliam no processo de interação dos agentes viabilizando diálogos para o alcance de objetivos.

3.2.5.1 *Contract net*

É um dos protocolos mais conhecidos no paradigma de orientação a agentes. Ele tem a finalidade de permitir que o agente inicie uma conversação enviando uma chamada de proposta *Call for Proposal* (CFP), para um conjunto de agentes respondedores. Assim, é feito um envio de proposta e a avaliação das propostas para então ser deliberado o aceite para qual o agente preferir (ou mesmo rejeitar todas). O protocolo de interação *Contract net* é especificado pela FIPA, sendo orientado a tarefas.

Nesse protocolo de interação, o *initiator* necessita de algum serviço, sabe o que deseja, mas não sabe exatamente quem vai prestar o serviço. Entretanto, ele tem consciência de que pode obter o melhor serviço se pesquisar e negociar. A Figura 3.10 ilustra esse protocolo. Note que o *initiator* envia uma mensagem CFP a m agentes *participants* fazendo um pedido de uma resposta de prestação de serviço, contendo todas as especificações e restrições do serviço que o agente precisa.

Em alguns casos os agentes *participants* que receberam a mensagem CFP não vão responder. Isso acontece porque o agente simplesmente ignorou a mensagem, não respondeu no tempo hábil definido pelo agente *initiator*, para o envio das respostas, ou simplesmente não tem interesse em executar a tarefa conforme as especificações e restrição do serviço.

Alguns agentes *participants*, em um conjunto j, que receberam a mensagem CFP e se "candidataram" a prover o serviço, enviam suas propostas (*propose*) ao *initiator*. O total de respostas é igual ao total de agentes que responderam à mensagem CFP, n, tirando o total de agentes que responderam *refuse*, recusando-se a enviar uma proposta ou que não entenderam a mensagem. Esta quantidade é representada na Figura 3.10 pela letra i.

Após o *initiator* receber propostas, vai escolher a melhor e informar aos agentes candidatos a sua decisão, enviando ao agente cuja proposta foi aceita uma mensagem *accept-proposal*, e *reject-proposal* aos agentes, cujas propostas não foram aceitas. O *initiator* pode utilizar mais de um serviço, pelo envio de várias mensagens *accept-proposal*.

O término da negociação é feita quando o *participant* que está ofertando o serviço recebe uma mensagem de *accept-proposal*. O agente vai realizar o que foi acertado, e depois envia uma mensagem de confirmação, informando que o serviço foi realizado como acertado (*inform-done*), e uma mensagem com o resultado (*inform-result*). Caso o agente que está executando o serviço não consiga realizar a tarefa, vai enviar uma mensagem *failure* e nela vai conter o motivo de não ter realizado a tarefa com sucesso.

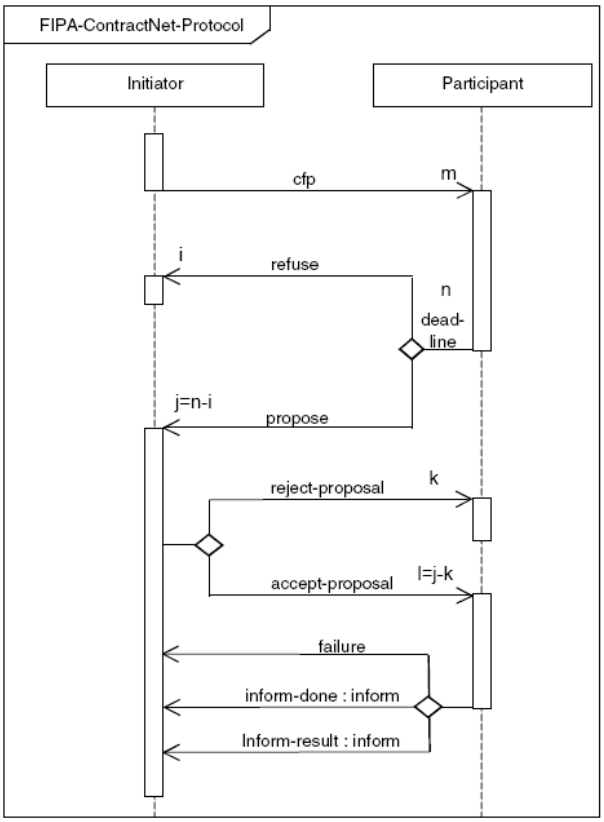


Figura 3.10: Protocolo de interação entre agentes FIPA Contract Net [40].

3.2.6 Arquiteturas

Uma arquitetura de agentes pode ser descrita como sendo um mecanismo fundamental, e a configuração dos componentes que constituem um sistema, e das conexões que coordenam as atividades entre esses componentes/agentes, fornecem o suporte ao comportamento efetivo no mundo real e ambiente dinâmico. O paradigma de agentes de software teve um esforço inicial no desenvolvimento de arquiteturas de agentes inteligentes [11]. Ao falar em arquitetura de agentes não estamos somente falando de arquitetura interna de cada agente, mas também da arquitetura do próprio SMA, ou seja, o modo como os agentes estão organizados dentro do sistema e também como estão estruturados os relacionamentos e interações.

Em [11, 151, 157], são apresentadas os quatro tipos principais de arquitetura: reativa baseada em raciocínio dedutivo, reativa baseada em crenças, desejos e intenções (*Believe Desire Intention - BDI*) e híbridas. Essas arquiteturas baseiam-se no paradigma de construção e interação dos agentes envolvidos [159].

3.2.6.1 Arquitetura Reativa

As arquiteturas reativas procuram utilizar modelo ou raciocínio baseado em IA clássica. Essa arquitetura tem por base a proposta de que um agente pode desenvolver inteligência a partir de interações com o seu ambiente, não necessitando de um modelo já pré-estabelecido [18]. Habitualmente o agente reativo toma suas decisões "em tempo

real", com base em um conjunto de regras de situação/ação, sendo esse conjunto de informações fixo e limitado, permitindo que selecione um dado comportamento. Segundo [11] não possuem modelos simbólicos ou de raciocínio dedutivo [11]. A informação proveniente dos sensores é em geral utilizada diretamente no processo de decisão, não sendo criada qualquer representação simbólica do mundo como ilustrado na Figura 3.11.

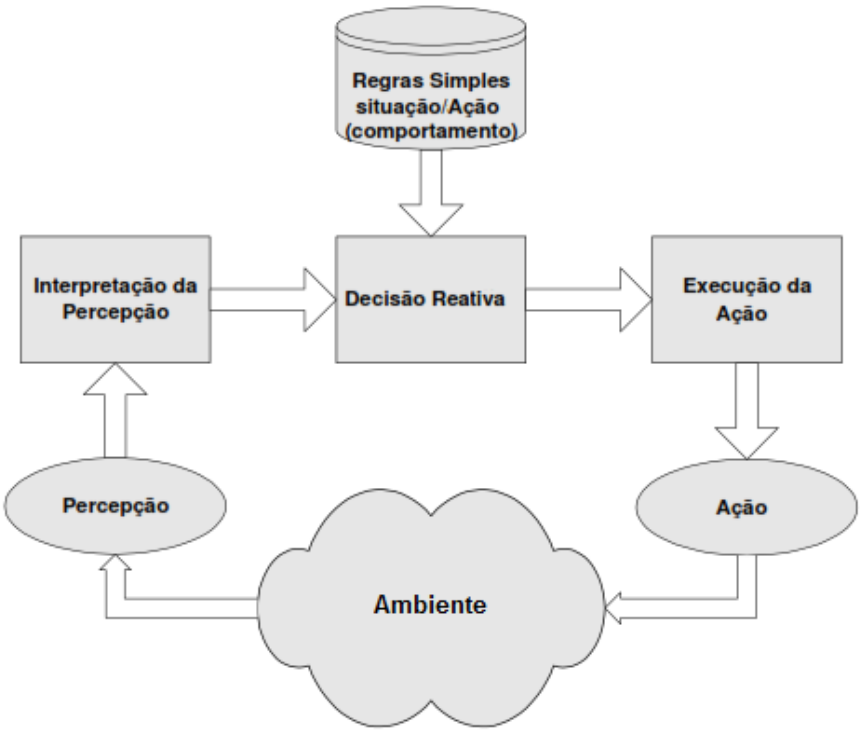


Figura 3.11: Esquema de uma arquitetura reativa (adaptado de [18]).

Essa arquitetura ficou conhecida como arquitetura de subsunção ou *subsumption architecture* [18]. O comportamento inteligente pode ser gerado sem uma representação explícita e o raciocínio abstrato do tipo de IA simbólica, sendo que a inteligência é uma propriedade emergente de alguns sistemas complexos. Existem duas características principais na arquitetura de subsunção, que são:

- o comportamento orientado a tarefa: agente deve tomar decisão através de um conjunto de comportamentos direcionados para a execução de tarefas, ou seja, cada comportamento deve ser pensado de forma individual de ação, que vai sendo tomada com base no conhecimento do ambiente e traduz essa percepção para uma ação a ser executada, como em uma máquina finita de estados;
- o acionamento simultâneo de comportamentos: o agente tem um mecanismo para selecionar entre as diferentes ações e escolhe aquela que será a ação ótima em cada momento. Desenvolve-se então uma hierarquia de subordinação com níveis de arquiteturas (*layers*), onde os comportamentos são organizados por níveis. Os níveis mais baixos são proprietários em relação aos níveis superiores, isso porque as ações de níveis mais altos representam comportamento mais abstratos.

A arquitetura reativa tem grandes vantagens: simplicidade, economia e boa robustez contra falhas [158]. Também tem desvantagens, tais como: no que se refere à característica de os agentes decidirem unicamente baseados em informação da percepção atual, assim, possuindo uma hierarquia de comportamento já pré-fixada e com isso são incapazes de realizar ações que impliquem a execução de planos de longo prazo.

3.2.6.2 Arquitetura Baseada em Raciocínio Dedutivo

As arquiteturas baseadas em lógica seguem a abordagem clássica de IA, onde os agentes atuam com raciocínio dedutivo e utilizam modelos simbólicos explícitos do ambiente, sendo os agentes conhecidos como agentes de raciocínio dedutivo. Estas arquiteturas interpretam em grande parte os agentes como parte de um sistema baseado em conhecimento [157].

As decisões dos agentes tem como base a manipulação pelo uso de mecanismo de formalização matemática, na qual a seleção de qual a próxima ação a executar é realizada através de raciocínio dedutivo. O agente possui uma representação interna do mundo e um estado mental explícito que pode ser modificado por alguma forma de raciocínio simbólico, sendo uma vantagem da arquitetura, pois o conhecimento humano é declarativo, facilitando a codificação e permitindo chegar a uma solução do problema [157].

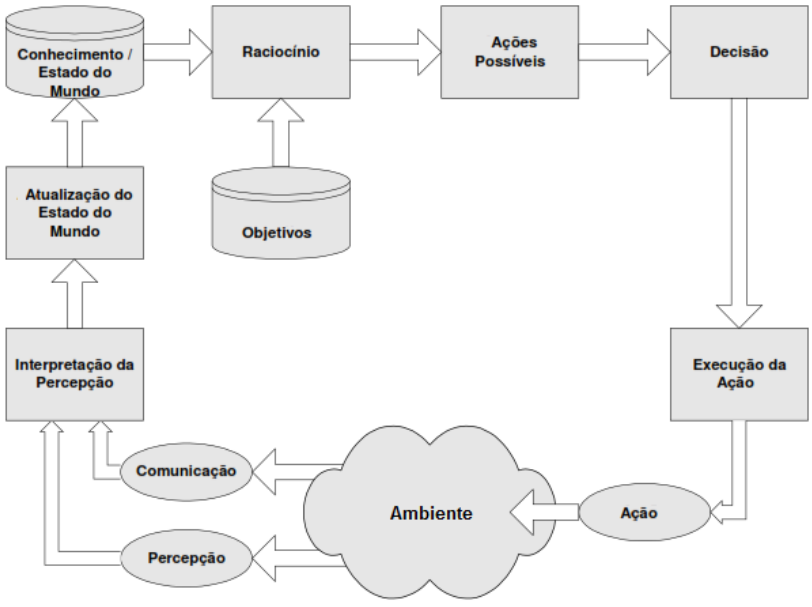


Figura 3.12: Esquema genérico de uma arquitetura baseadas em raciocínio dedutivo (adaptado de [18]).

Entretanto, na utilização e construção de agentes seguindo esta arquitetura, colocam-se dois importantes problemas/desvantagens: o problema de como traduzir o mundo real para uma descrição simbólica e utilizar a percepção dos agentes para manter essa estrutura simbólica atualizada. Outro problema é como raciocinar utilizando essa informação simbólica para decidir sobre as ações a serem executadas em cada momento específico [157].

A Figura 3.12 apresenta um esquema típico de uma arquiteturas baseadas em lógica. Depois da interpretação da percepção do ambiente, o agente vai utilizar essa informação

para atualizar suas representações internas, usualmente simbólica, mostrando o estado do mundo. O agente vai utilizar o estado do mundo em conjunto com os objetivos do agente para gerar possíveis ações a serem executadas pelo agente, e com base nisso, selecionar as mais apropriadas a executar pelo agente.

3.2.6.3 Arquitetura baseada em crenças, desejos e intenções

A arquitetura baseada em crenças, desejos e intenção (*Belief-Desire-Intention* BDI) [15] é uma arquitetura essencialmente deliberativa baseada no modelo intencional humano, onde o agente é descrito através de um conjunto de "estados mentais", sendo estes o estado interno de processamento. No entanto, as noções de crença ("*belief*"), desejo ("*desire*") e intenção ("*intention*") são aceites unanimemente usando raciocínio teórico. Essa arquitetura foi denominada como raciocínio prático, por ser direcionada às ações, portanto, ás condições de como o agente deve descobrir, o que fazer, e como fazer (raciocínio meio-fim) [15].

Com base no raciocínio prático, o agente faz um balanço entre suas considerações conflitantes, tanto a favor como contra, sendo as considerações relevantes levadas em consideração incluindo os desejos/valores com base no que o agente crê ser verdade.

As crenças de um agente referem-se ao que o agente acredita ser verdade em cada instante no ambiente, descrevendo o estado do mundo do agente no exato momento, sendo a base de crenças formada por informações do mundo ou ambiente.

Os desejos de um agente referem-se ao que o agente deseja obter. Entretanto, nem sempre o agente tem as formas de alcançar esses desejos em um dado instante, tomando-os inconsistentes em um dado momento.

Os objetivos/intenções/desejos resultam em um processo de raciocínio, que consiste em uma escolha de um subconjunto dos desejos que são consistentes e acessível por parte do agente. As intenções tem como base um conjunto de ações ou tarefas selecionadas pelo agente, impactando assim na realização dos seus objetivos. Com base em intenções consistentes, o agente vai representar o resultado deliberado pelo processo.

A arquitetura genérica de um agente BDI proposta por [151] é constituída por sete componentes: conjunto de crenças, função de atualização das crenças, função de geração de opções, conjunto de opções, de desejos, função de filtragem, conjunto de intenções e função de seleção de ação, como ilustrado na Figura 3.13 e a interpretação percepção e execução da ação.

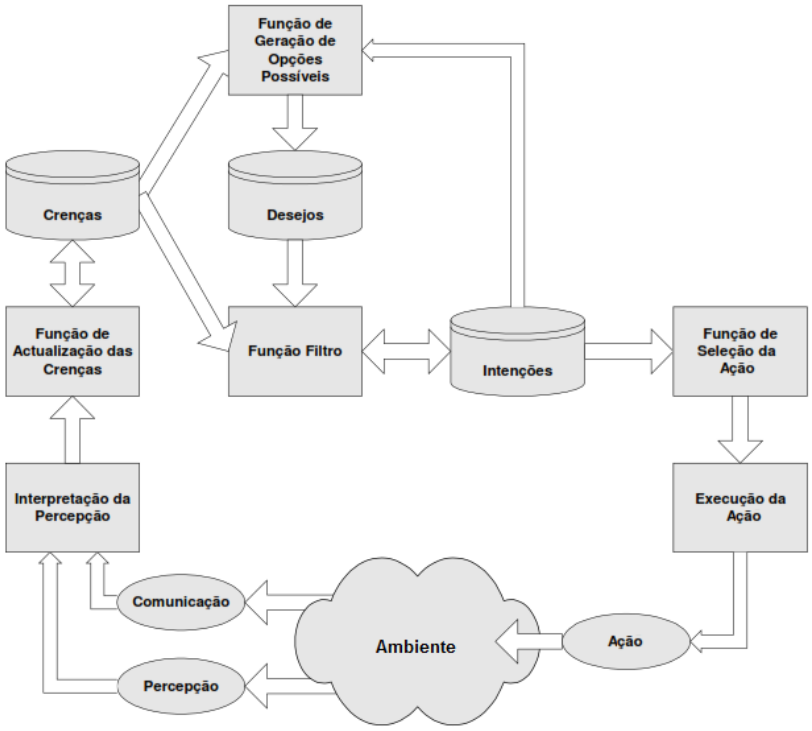


Figura 3.13: Esquema genérico da arquitetura BDI (adaptado de [151]).

3.2.6.4 Arquiteturas Híbridas

A arquitetura hierárquica em camadas, como o próprio nome indica, compreende a existência de vários subsistemas que se organizam em hierarquias e interagem por níveis. Essa arquitetura também é conhecida como arquitetura híbrida, que permite ambos os comportamentos para seus agentes, o comportamento reativo e deliberativo. Essa arquitetura leva em consideração dois tipos de camadas interativas conforme o fluxo de controle: fluxo de controle horizontal e vertical [157].

Na arquitetura em camada horizontal, as camadas de software estão diretamente ligadas aos sensores de *entrada* e às ações de *saída*, ou seja, cada camada vai atuar com um ou mais agentes, uma vez que vai apresentar sugestões de ação para o(s) agente(s) (Figura 3.14).

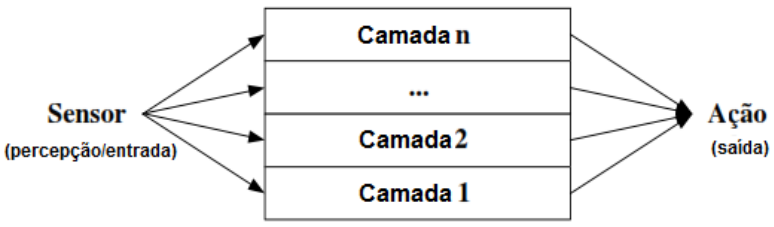


Figura 3.14: Arquitetura em camadas horizontais com fluxo de controle (adaptado de [157]).

Na arquitetura em camada vertical, os sensores de *entrada* e às ações de *saída* têm no mínimo uma camada entre eles, isto é, têm de ser trabalhado, pelo menos, por uma camada (Figura 3.15), sendo que o controle pode ser feito em um único sentido (1) ou duplo (2).

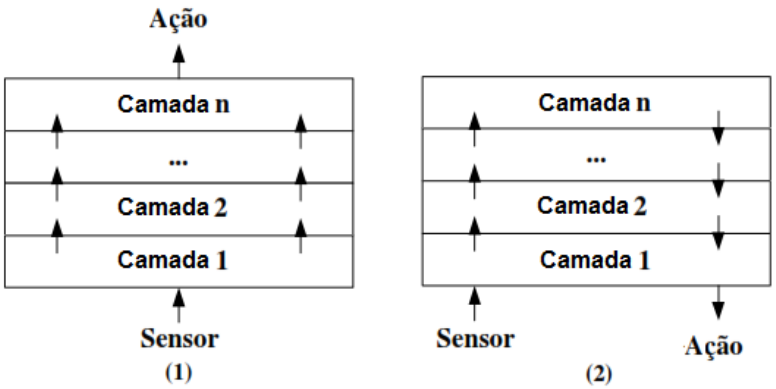


Figura 3.15: Arquitetura em camadas verticais com controle único e duplo (adaptado de [157])).

Comparando as arquiteturas em camadas, considere n camadas arquiteturais e cada uma capaz de sugerir m ações, significa que teremos m^n interações para considerar. Essa dificuldade é amenizada com arquiteturas verticais. Por outro lado, uma vez que as camadas competem umas com as outras, poderá ser necessária a introdução de um mediador para auxiliar na tomada de decisão.

Já na arquitetura vertical o fluxo de controle passa de maneira sequencial por cada camada, até que todas as camadas gerem uma ação que vai ser executada. Entretanto, existe nessa arquitetura pouca flexibilidade, já que para o agente tomar uma decisão é preciso que o fluxo de controle passe por todas as camadas, sendo assim, caso exista uma falha em alguma das camadas, o agente terá problemas para gerar e executar a ação.

3.2.7 Ferramentas

Nesta seção, inicialmente descreveremos ferramentas para a construção de SMAs e em seguida faremos uma comparação entre elas.

3.2.7.1 Descrição das Ferramentas

JACK

É um ambiente de desenvolvimento orientado a agentes desenvolvido pelo Grupo de AOS[3]. JACK [154] foi desenvolvido para fornecer extensões orientada à agente utilizando a linguagem de programação Java. Foi concebido como um conjunto de componentes com alta performance e forte tipagem de dados. JACK trabalha com o paradigma BDI. Os agentes são definidos em termos de suas crenças (sabem o que tem que fazer e como fazer), os seus desejos (quais os objetivos que gostaria de alcançar), e suas intenções (os objetivos

[3]Agent Oriented Software

que estão realmente empenhados em alcançar). Tem um ambiente de desenvolvimento gráfico que suporta implementação e o monitoramento de agentes BDI, pode ser executado em multi-plataforma.

JADE

O JADE (*Java Agent DEvelopment Framework*) [10] segue os padrões da FIPA. O JADE possibilita e facilita a programação de agentes inteligentes usando Java, possui muitos atributos e características para a implementação de agentes (transporte de mensagens, páginas brancas e amarelas, protocolos de comunicação e de interação), além de simplificar o desenvolvimento por disponibilizar um *framework* que trata a comunicação, o ciclo de vida do agente e o monitoramento da execução, com diversos comportamentos pré-definidos e modelos de interação com métodos implementados.

É uma ferramenta multi-plataforma e trabalha com agentes distribuídos. Fornece os serviços de infra-estrutura de comunicação, tais como gerenciamento de agente e um conjunto de ferramentas de desenvolvimento e depuração. É um *middleware* que garante as especificações relacionadas com o FIPA no quesito interoperabilidade, segurança e manutenção. É completamente implementado em Java sendo distribuído com licença LGPL-2[4].

Jadex

É um pacote de extensão do JADE e é uma implementação de uma arquitetura híbrida com agentes (reativos e deliberativos), construído para permitir o desenvolvimento de agentes de acordo com a FIPA e a arquitetura BDI. JADEX [114] permite a construção de agentes de software explorando o modelo BDI:

i) Usa tecnologias como XML e Java;

ii) JADEX é projetado para facilitar a implementação de agentes em Java, e, portanto permite o reuso de várias ferramentas e bibliotecas.

Um agente JADEX é também um agente JADE e, portanto todas as ferramentas disponíveis em JADE podem ser usadas para desenvolver agentes em JADEX. A maior parte da plataforma JADE lida com a visão externa de um agente, que não difere entre um agente JADE ou JADEX.

Jason

O Jason [13] é uma ferramenta de código aberto de uma versão estendida do *AgentSpeak*, uma linguagem de programação orientada agente. É escrita baseada em Java. Ele permite aos usuários construir SMA complexo. Jason é adequado para a implementação de sistemas reativos de acordo com a arquitetura BDI. Jason usa as bibliotecas da FIPA para a comunicação entre agentes e é capaz de trabalhar com o JADE. Ele está disponível Open Source, sendo distribuído sob licença GNU LGPL.

[4]GNU General Public License (Licença Pública Geral), GNU GPL ou simplesmente GPL, é a designação da licença para software livre

SimAgent

É um conjunto de ferramentas que foi desenvolvido por Aaron Sloman, da Universidade de Birmingham. Ele foi projetado para implementar e testar arquiteturas de diferentes agentes, incluindo cenários onde cada agente é composto por vários tipos diferentes de subsistemas concorrentes, que interagem em um ambiente onde exite outros agentes. Ela foi originalmente desenvolvida para apoiar a pesquisa sobre agentes inteligentes semelhantes a humanos, mas também tem sido usado para projetos de desenvolvimento de jogos e simulações. Alguns agentes possuem sensores e alguns precisam de autorização para comunicar-se com os outros. Alguns agentes podem ter arquiteturas híbridas. Usa a linguagem Pop-11 no ambiente de desenvolvimento de software Poplog.

Zeus

É um ambiente integrado para a construção de aplicações com agentes colaborativos. A documentação da ferramenta Zeus [106] coloca uma forte ênfase na metodologia de Zeus. Os três principais componentes do ZEUS são: a biblioteca de componentes para agentes, ferramentas de contrução de agentes e as ferramentas de visualização. A ferramenta Zeus foi desenvolvida pela *British Telecommunications* (BT), e fornece uma biblioteca de componentes de software e ferramentas que facilitam o desenho, desenvolvimento e implantação de SMA. A metodologia de Zeus usa a decomposição de quatro estágios para o desenvolvimento de agente, respectivamente análise de domínio, design, realização e apoio a execução.

3.2.7.2 Análise dos Critérios de Avaliação

Os critérios de avaliação selecionados estão diretamente relacionados com o objetivo de facilidade de desenvolvimento de um SMA. Obviamente, as ferramentas avaliadas não contemplam a totalidade de opções existentes na literatura.

Como critérios especifico de avaliação foram utilizados:

1. é *open source* distribuído sob licença GNU?
2. é uma ferramenta multiplataforma?
3. permite utilizar modelo de raciocínio dedutivo?
4. é FIPA *compliance*?
5. Tem métodos para uso protocolo de interação *contratc-net*?
6. Tem protocolo de comunicação FIPA ACL?

Com base na Tabela 3.1, pode-se verificar que a ferramenta JADE conseguiu o melhor resultado na avaliação, sendo utilizada neste trabalho.

3.3 Agentes com Raciocínio Dedutível

Como estamos interessados no desenvolvimento de um SMA cognitivo, o mecanismo de raciocínio dos agentes é fundamental. Foi realizado um levantamento de ferramentas *shell* para raciocínio dedutivo dos agentes.

Tabela 3.1: Avaliação das Ferramentas

	Critérios					
Ferramentas	**1**	**2**	**3**	**4**	**5**	**6**
JACK		×	×		×	×
JADE	×	×	×	×	×	×
Jadex	×	×	×	×	×	×
Jason	×	×		×	×	
SimAgent	×		×			
Zeus	×	×		×	×	×

Para implementar um raciocínio dedutivo nos agentes faz-se necessário definir regras de comportamento. Neste trabalho foi utilizado uma base de conhecimento para os agentes com um motor de inferência para executar as regras de produção.

Um motor de inferência baseado em regras tem mecanismo de encadeamento progressivo (*forward chaining*), e encadeamento regressivo (*backward chaining*) [122]. O encadeamento progressivo, também é conhecido como encadeamento dirigido por fatos. O encadeamento regressivo, também é conhecido como encadeamento dirigido por objetivos, o comportamento do sistema é controlado por uma lista de objetivos. Esses objetivo pode ser satisfeito por um elemento que está na memória de trabalho, ou pela existência de regras que permitem a inferência deste objetivo.

As regras que satisfazem a condição têm a instância correspondente a sua parte da esquerda adicionada a lista de objetivos correntes. Quando uma destas regras tem suas condições satisfeitas diretamente pela memória de trabalho, o objetivo em sua parte direita também é adicionado a memória de trabalho. Se um objetivo não é satisfeito diretamente pela memória de trabalho, e também não é inferido através de uma regra, não é acionado. Quando o objetivo inicial é satisfeito, ou não há mais objetivos, o processamento é interrompido.

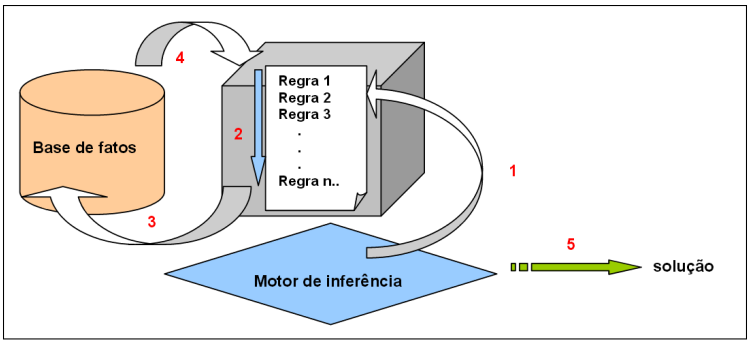


Figura 3.16: Ciclo de Operações em um Motor de Inferência (adaptado de [122]).

Uma estratégia de busca é utilizada para guiar e facilitar a pesquisa na memória trabalho e na base de regras/fatos. Uma resolução de conflitos é executada após o término da busca quando o motor de inferência possui um conjunto de regras que satisfaça as premissas atuais do problema, então, o conjunto recebe o nome de conjunto de conflito. O motor de inferência deve escolher as regras que serão executadas e a ordem da execução destas regras. A quantificação de incertezas são atribuídos valores de pesos quantitativos

aos fatos e as regras. A Figura 3.16 mostra o processo de operação realizado pelo motor de inferências, do recebimento da solicitação para percorrer a base de regras, gravar novos fatos e enviar resposta final.

3.3.1 Motores de Inferência

Existem na literatura várias implementações de motor de inferência, dentre estas eles o JESS (*Java Expert System Shell*) [43], Drools [21], *Hammurapi Rules* [52] e *JRuleEngine* [66], os quais foram avaliados neste trabalho.

JESS

O JESS [43], criado pela NASA em 1997, teve como objetivo a integração da abordagem Orientado a Objetos (OO) com regras de produção de domínios de conhecimento. O JESS é um *shell* para criação de sistemas baseados em regras e foi utilizado em sistemas construídos com as linguagens de programação C e C++.

Seu motor de inferência utiliza o algoritmo RETE[5] [42]. O JESS é na verdade um *shell*, no qual os comandos são utilizados em uma linguagem própria (linguagem de regras *Jess* ou em XML) que, além de permitir a criação de *scripts* de regras de interpretação, possibilita que os objetos Java sejam utilizados como ligações para os fatos do contexto.

Drools

O Drools [21] é uma ferramenta para construção de bases de conhecimento e de inferência dirigida por padrões. Foi construído para interagir com Java e o conhecimento é obtido de regras declarativas. Uma regra Drools tem uma ou mais condições (ou fatos) que levam a uma ou mais ações (ou consequências).

Basicamente, o motor de inferência do Drools oferece a possibilidade de utilização do método de encadeamento progressivo e regressivo. O algoritmo de inferência presente no Drools é o RETE [42]. Como no JESS é adaptado para sistemas orientados a objetos. JBoss possui uma plataforma de lógica de negócio chamada JBoss Drools, onde pode ser acoplado o motor de inferência. O sistema teve sua primeira versão lançada em 2001, estando na versão 6.1.0.

O *Drools* é composto por: (i) uma máquina de inferência, que é responsável pela execução das regras; (ii) uma memoria de trabalho, utilizada para armazenar os fatos gerados pela execução das regras; (iii) uma base de conhecimento, que é o local onde estão contidas as regras que o mecanismo de inferência vai utilizar. Na Figura 3.17 é mostrado o diagrama com os elementos que compõem o Drools.

Com o Drools é possível elaborar regras de negócio declarativas, separar e centralizar as regras de negócio de uma aplicação, e fazer o gerenciamento das regras alterando-as e mudando suas versões dinamicamente.

[5]Algoritmo de Markov responsável pela especificação de uma estrutura de controle para sistemas especialistas baseado em regras de produção. Possui um grupo ordenado de regras de produção, as quais são aplicadas na ordem de prioridade dos textos de entrada.

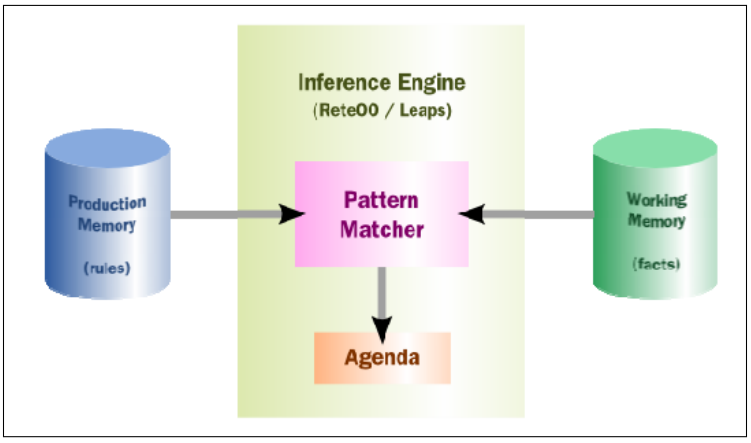


Figura 3.17: Diagrama com os elementos que compõem o Drools [21].

Hammurapi Rules

Hammurapi Rules [52] é um software desenvolvido em Java, cuja principal idéia, segundo seus desenvolvedores (*Hammurapi Group*), é a de apoiar o desenvolvimento de sistemas baseados em regras de uma maneira mais simples e rápida. O motor de inferência do *Hammurapi Rules* oferece a possibilidade de utilização dos dois métodos de encadeamento das regras, o encadeamento progressivo e o encadeamento regressivo, com a implementação do tradicional algoritmo de inferência RETE [42], como no JESS e no Drools.

O Hammurapi Rules permite utilizar a própria linguagem Java para escrita das regras e, consequentemente, os próprios objetos do domínio. No Hammurapi Rules os operadores presentes na linguagem Java são responsáveis por descrever as relações e condições presentes nas regras.

JRuleEngine

JRuleEngine [66] é um software desenvolvido em Java, com base na especificação JSR-94 *(Java Specification Request 94)*, versão 2.1. Nesse mecanismo os objetos de entrada são referidos como fatos e os objetos de saída são referidos como conclusões. O *JRuleEngine* permite chamar métodos de uma determinada classe diretamente das suas regras. Ele usa um motor de inferência baseado em encadeamento progressivo.

Sua arquitetura é composta de uma memória de trabalho que contém os fatos, que foram gerados pela execução das regras do sistema, possui também uma base de conhecimento, onde as regras utilizadas para a inferência estão armazenadas, e o mecanismo de inferência, responsável pelo processamento das regras, acesso a métodos externos e alimentação da base de fatos, como podemos observar na Figura 3.18 .

3.3.1.1 Definição e Análise dos Critérios de Avaliação

Neste trabalho foi considerado para integração com a linguagem de programação:

- presente se o software ou ferramenta oferecer uma maneira de implementar as regras por meio dos próprios objetos Java, sem *scripts* intermediários.

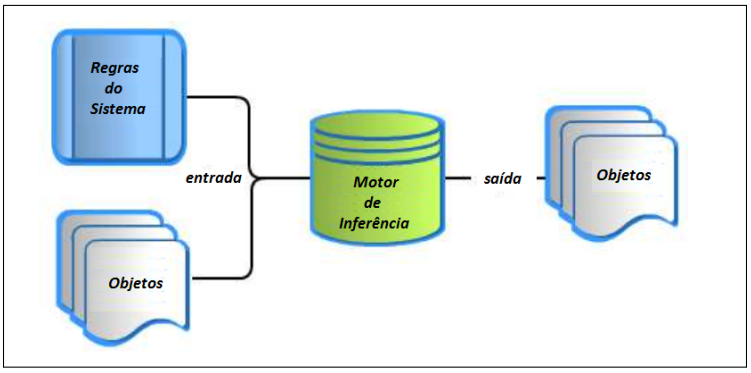


Figura 3.18: Arquitetura JRuleEngine [66].

- parcial, o produto oferece uma maneira de implementar as regras por meio dos próprios objetos JAVA, com scripts intermediários.
- ausente o caso do produto não oferecer nenhuma maneira de implementar as regras por meio dos próprios objetos Java.

Além disso, para a monitoração do contexto foi considerado:

- presente se o produto oferecer as duas maneiras mencionadas de monitoração automática, sendo: monitoração periódica e monitoração por eventos.
- parcial se o produto oferecer pelo menos uma das duas maneiras mencionadas de monitoração automática - monitoração periódica ou monitoração por eventos.
- ausente, o caso do produto não oferecer pelo menos uma das duas maneiras mencionadas de monitoração automática - monitoração periódica ou monitoração por eventos.

Os critérios de avaliação foram estabelecidos para formalizar de maneira adequada o raciocínio dos biólogos no ncRNA-Agents. A análise desses critérios foi definida em termos de:

- ausência do recurso: AUSENTE;
- presença parcial do recurso: PARCIALMENTE; e
- presença completa do recurso: PRESENTE.

Os critérios avaliados nas ferramentas foram:

1. Integração com a linguagem de programação: Os quatro produtos apresentam algum tipo de relação com a linguagem Java. É passível de uma explicação o fato de que o JESS não tenham obtido a classificação máxima "PRESENTE"). Essa ferramenta têm o propósitos um pouco mais geral do que o as outras ferramentas. Em vista disso, o JESS abdica de um compromisso maior com a integração, buscando uma maior abrangência de apoio no desenvolvimento. O JESS auxilia a criação de sistemas baseados em regras em Java, mas só a experiência na linguagem de programação não é suficiente para o processo de desenvolvimento.

2. Monitoração automática do contexto: No JESS, a monitoração do contexto está presente, por meio da relação entre os objetos do domínio e o contexto. O JESS oferece uma monitoração automática com base na abordagem de "monitoração por eventos", na qual sempre que os objetos do domínio alteram, as regras são investigadas. No caso do JESS, todas as regras são investigadas sempre que um dos objetos do domínio sofre alteração. No Drools, a monitoração automática pode vir a ter melhor desempenho de predição de ncRNA que o JESS, mesmo adotando a abordagem de "monitoração periódica". O número de regras e o domínio do sistema serão determinantes para uma possível medida de desempenho entre as duas abordagens.

3. Expressão de regras de primeira ordem: Desconsiderando o número de operadores, pode-se afirmar que a expressão do conhecimento por meio de regras de primeira ordem está presente nos quatro produtos.

4. Expressão de regras de segunda ordem[6] ou temporais: As regras de segunda ordem ou temporais só estão presentes no Drools.

A Tabela 3.2 traz uma comparação entre os motores de inferência, mostrando o *Drools* com o melhor resultado para os critérios propostos. O *Drools* possui algumas vantagens adicionais além da capacidade normal de um motor de inferência. O *Drools* tem uma interface amigável para o usuário com um grande conjunto de ferramentas de apoio, tais como o *Guvnor*, que é um sistema de gerenciamento de regras de negócio (SGRN), que tem controle de versão e gestão.

Tabela 3.2: Avaliação dos Motores de Inferência.

Critérios	**Produtos**			
	JESS	**Hummurapi Rules**	**Drools**	**JRuleEngine**
Integração com linguagem de programação	Parcialmente	Presente	Presente	Presente
Monitoração automática do contexto	Presente	Ausente	Presente	Presente
Expressão de regras de primeira ordem	Presente	Presente	Presente	Presente
Expressão de regras temporais	Ausente	Ausente	Presente	Ausente

3.4 Tipos de Agentes, Arquitetura e Ferramentas usadas nesta Tese

Desta forma, os agentes e a arquitetura, apresentados na Figura 4.1 têm as seguintes características:

[6] é uma extensão da lógica de primeira ordem com adição de variáveis e quantificadores sobre conjuntos de indivíduos.

- Agentes reativos e cognitivos com raciocínio dedutível; e
- Arquitetura reativa, em camadas verticais com controle duplo.

As ferramentas utilizadas são:

- JADE com protocolos de interação *Contract-Net* e Linguagem de comunicação FIPA-ACL; e
- *Drools* utilizada como motor de inferência para simular o raciocínio dos biólogos.

Capítulo 4

ncRNA-Agents

Neste capítulo, apresentamos a arquitetura do sistema de anotação de ncRNAs baseado em SMAs. Na Seção 4.1, é mostrado uma classificação dos métodos computacionais descritos no Capítulo 2, que será utilizada na arquitetura do SMA proposto nesta tese. Na Seção 4.2, será descrito o raciocínio utilizado para decidir a anotação de uma sequência de DNA ou RNA, que simula um anotador humano. Na Seção 4.3, será descrito a arquitetura que usa agentes colaborativos. Por fim, na Seção 4.4, serão abordados os detalhes da implementação do sistema ncRNA-Agents.

4.1 Classificação de Métodos Computacionais

Propomos uma classificação dos métodos computacionais para anotação de ncRNAs em três grupos [48], como segue:

Homologia

Esta classe de ferramentas envolve predição de ncRNAs feita por meio de comparação de sequências entre duas ou mais espécies. Essas comparações dependem de bancos de dados curados, no sentido de que quanto melhores forem as anotações do banco melhores serão as predições [86].

Dois genes são ditos homólogos se descendem de um ancestral comum, e possivelmente esses genes mantenham a mesma funcionalidade herdada. Sequências homólogas podem ser divididas em duas classes: ortólogas e parálogas. As ortólogas são sequências relacionadas por especiação, possuindo uma descendência vertical, já as parálogas são sequências relacionadas por duplicação dentro da mesma espécie ou nos ancestrais [138].

Duas ferramentas importantes para inferir homologia, que usam como métrica similaridade de sequências (quanto maior a similaridade entre duas sequências, maior a chance delas serem homólogas), são: BLAST [1], uma ferramenta para detecção de ncRNAs baseada em alinhamento par a par entre as estruturas primárias das sequências; e o Infernal [1], uma ferramenta baseada em alinhamento múltiplo que considera as estruturas secundárias das sequências. O BLAST, que anota proteínas de forma confiável, em geral não produz bons resultados para anotar ncRNAs. Porém, o Infernal é mais sensível e específico, pois foi projetado especialmente para anotar ncRNAs [34].

Predição de Classe

Esta classe inclui ferramentas de anotação de ncRNAs baseada em métodos de aprendizagem de máquina.

No aprendizado supervisionado, pode-se tomar um conjunto conhecido de ncRNAs e um conjunto conhecido de proteínas, calculando características *ab initio* dessas sequências, visando criar um modelo de predição de ncRNAs. Isso confere ao modelo maior confiabilidade.

Por exemplo, o SVM-PORTRAIT [5, 6] mostra um grau de acurácia elevado para sequências de transcritomas ainda não totalmente caracterizadas.

Modelos *De Novo*

Essa classe inclui ferramentas para anotação de ncRNAs criadas a partir de outros modelos, diferentes de homologia e predição de classe.

Por exemplo, no modelo termodinâmico, a ordem dos nucleotídeos na sequência primária e as possibilidades de pareamento em uma molécula de RNA de tal forma a diminuir a energia livre da estrutura são usadas para predizer sua conformação espacial. Uma investigação dessa conformação, por sua vez, resulta em um conhecimento aproximado sobre suas propriedades fisiológicas [162].

4.2 Processo de Raciocínio Baseado em Conhecimento

No intuito de anotar uma sequência, os biólogos combinam resultados de diversas ferramentas computacionais utilizando seu conhecimento biológico. Esse raciocínio pode ser modelado da forma descrita a seguir.

Em primeiro lugar, uma vez que os resultados de diferentes ferramentas fornecem informações sobre vários aspectos de uma sequência (essa é a motivação para o agrupamento de ferramentas computacionais em classes), os resultados obtidos de cada uma delas pode aprimorar bastante a anotação.

Para começar, os resultados do grupo de ferramentas da classe de homologia vão pesquisar a estrutura primária da sequência investigada: tRNAscan verifica se a sequência é um tRNA; Blast sugere uma anotação identificando alinhamentos bons em relação às sequências das suas bases de dados; Infernal pode recomendar tRNA, como tRNAscan, ou a mesma anotação do Blast, ou ainda dar outras sugestões. Um anotador humano analisa todos esses resultados e toma uma decisão de qual é a melhor anotação para a sequência que está analisando.

A seguir, os resultados das ferramentas relacionadas ao grupo de predição de classe, por exemplo, SVM-Portrait (essa ferramenta vai computar a probabilidade da sequência ser um ncRNA) pode confirmar a sugestão inferida pelas ferramentas que estão no grupo de homologia, ou não.

No entanto, uma outra ferramenta, por exemplo, o RNAfold (pacote do Viena) mostra se a sequência investigada tem conformação espacial de um ncRNA, ou não.

Se as ferramentas de cada grupo sugerem a mesma anotação, a anotação recomendado para a sequência é considerada "boa". Caso contrário, se as ferramentas sugerem anotações que concordam apenas parcialmente, ou se apenas parte das ferramentas recomendam

uma anotação, esses resultados vão ser analisado e combinado, considerando o grau de confiabilidade de cada ferramenta, o que vai ser determinado para cada projeto.

Além disso, o anotador humano pode avaliar se a anotação inferida é "confiável", verificando se a anotação predita é a mesma ou próxima a de organismos próximos filogeneticamente. Assim, assumimos que conservação de sequências entre organismos relacionados reforça a confiabilidade da anotação predita de ncRNAs.

Porém, se as ferramentas não sugerem uma anotação "boa" nem "confiável", a busca de sequências similares, anotadas como ncRNAs em organismos relacionados, vai sugerir uma anotação de "*ncRNA novo*" para a sequência que está sendo investigada.

Em resumo, a Tabela 4.1 mostra as três indicações de qualidade de anotação, criadas a partir das classes de ferramentas propostas.

Tabela 4.1: Avaliação da qualidade da anotação

Qualidade da anotação	**Razão**
boa	Anotação por homologia, confirmada por predição de classe e/ou *de novo*.
confiável	Anotação por homologia, confirmada por predição de classe e/ou *de novo* e apresentando conservação entre organismos relacionados.
ncRNA novo	Inferida apenas a partir da conservação entre organismos relacionados.

4.3 Arquitetura

No presente projeto, inspirado em trabalhos anteriores de Ralha e co-autores [117, 127], um sistema para anotação baseado em SMAs, denominado ncRNA-Agents é apresentado. A arquitetura apresenta quatro diferentes camadas, conforme apresentado na Figura 4.1.

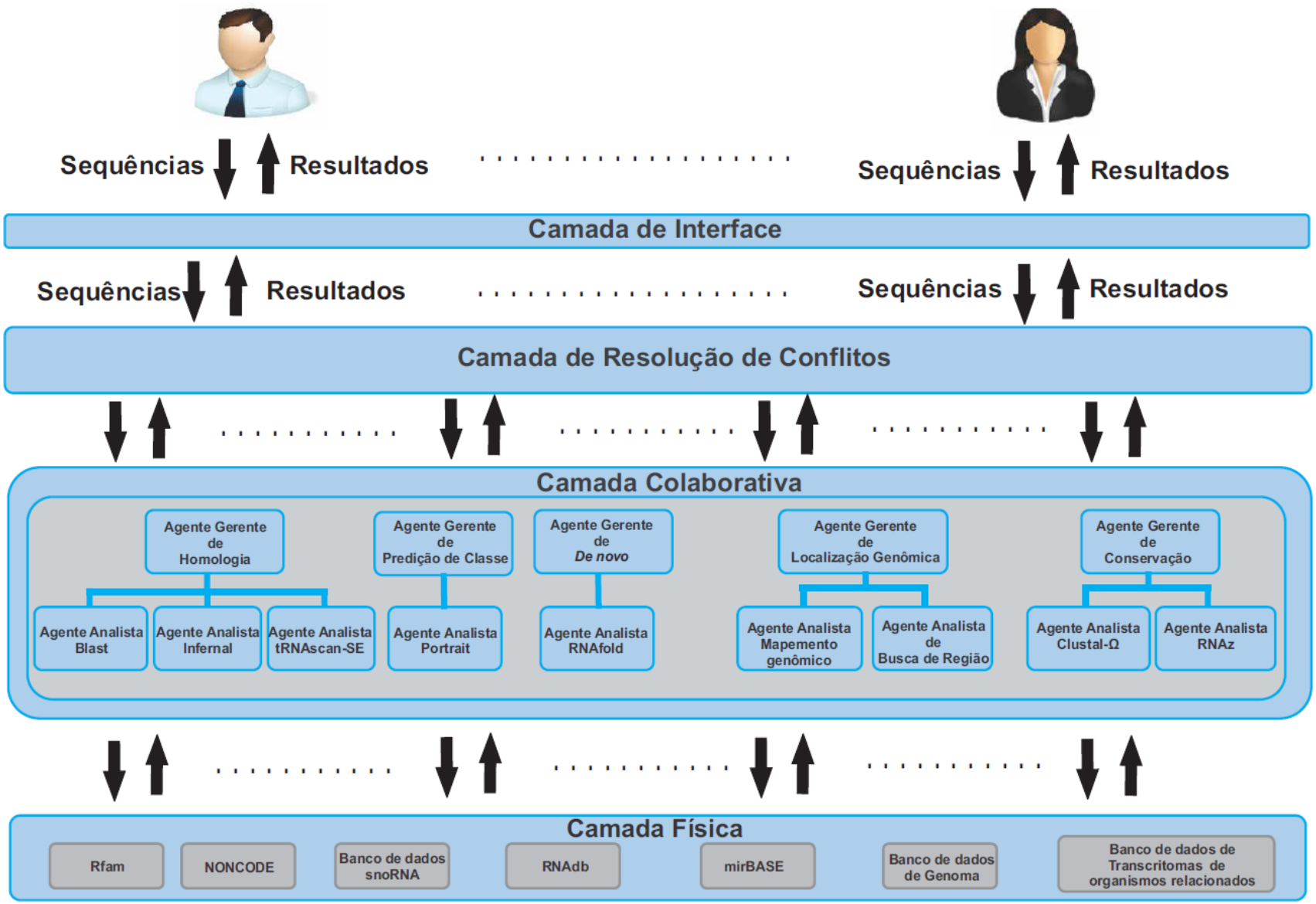


Figura 4.1: Arquitetura do ncRNAs-Agents.

4.3.1 Descrição das Camadas

A **camada de interface** recebe do usuário uma requisição, composta por um sequência, ou um arquivo com sequências, em formato FASTA, além da indicação de um conjunto de ferramentas de anotação. Após o processamento, esta camada retorna, para cada sequência, a anotação, predita como ncRNA (se for o caso), juntamente com os resultados e cálculos feitos pelas ferramentas, que foram utilizadas no processo de raciocínio.

A **camada de resolução de conflitos** decide qual é a melhor recomendação para a anotação de cada sequência, a partir das diversas sugestões recebidas da camada colaborativa. Nesta camada, é simulado o raciocínio do biólogo, que informa essa decisão para a camada de interface.

A **camada colaborativa** é responsável pela execução das diversas ferramentas (escolhidas pelo usuário) para anotar ncRNAs, e pela filtragem dos resultados das ferramentas. Em seguida envia os resultados filtrados para a camada de resolução de conflitos.

A **camada física** é formada por banco de dados.

4.3.2 Descrição dos Agentes

Na **camada colaborativa** existem dois tipos de agentes: **Agentes Gerentes** e **Agentes Analistas**. Os **Agentes Gerentes** realizam um filtro nas sugestões enviadas pelos Agentes Analistas. Temos cinco tipos de Agentes Gerentes, três deles baseados nas três classes propostas na Seção 4.1, além de, dois outros gerentes criados para avaliar a anotação recomendada (conforme descrito na seção 4.2). Esses dois agentes visam remover falsos positivos, ou procurar ncNRAs novos, encontrados a partir da conservação da sequência em organismos relacionados. Os agentes gerentes também simulam o raciocínio dos biólogos.

O **Agente Gerente de Homologia** coordena agentes que trabalham com ferramentas baseadas em similaridade. Dois exemplos são: (i) BLAST [1], que considera apenas a estrutura primária da sequência mas identifica bem snoRNAs [32]; e (ii) Infernal [34], que considera a estrutura secundária da sequência.

O **Agente Gerente de Predição de Classe** coordena os agentes que trabalham com ferramentas baseadas em Aprendizagem de Máquina. Um exemplo é o SVM-Portrait [5].

O **Agente Gerente *De novo*** gerencia agentes que trabalham com ferramentas que não usam homologia nem predição de classe. Um exemplo é o RNAz [149], do pacote do Vienna, baseado em modelo termodinâmico.

O **Agente Gerente de Localização Genômica** coordena agentes que trabalham com ferramentas de alinhamento, resultantes de métodos de comparação de sequências, com o intuito de descobrir se uma sequências pode ser mapeada em um genoma de referência o que, permite verificar se a sequência e de fato um ncRNA. Sequência não mapeadas são consideradas falso positivos. Por exemplo, pode-se buscar as posições em um genoma de referência, se a sequência for mapeada nele; ou pode-se localizar a sequência em exons, introns ou regiões intergênicas, se essa informação estiver disponível.

O **Agente Gerente de Conservação** coordena agentes que trabalham com ferramentas que investigam conservação entre os organismos relacionados.

Finalmente, os **Agentes Analistas** são responsáveis por executar ferramentas específicas para anotação de ncRNAs. Cada Agente Analista, criado por solicitação de um agente gerente, executa uma análise (*parse*) para extrair informações do arquivo de

saída criado pela ferramenta específica controlada por ele. O resultado dessa análise é retornado ao Agente Gerente solicitante como recomendação de anotação. O Gerente de Conservação, que investiga domínios conservados entre os organismos relacionados, utiliza dois Agentes Analistas, um para realizar o alinhamento múltiplo (por exemplo, usando Clustal-Ω), que é a entrada para um outro agente que vai calcular a probabilidade da estrutura do RNA ser conservada (por exemplo, usando o RNAz).

Para o alinhamento múltiplo, é utilizada a sequência que está sendo investigada, que é comparada com genomas (cromossomo, *scaffolds*, *supercontigs* ou *contigs*) dos organismos, relacionados as sequências das regiões similares.

4.4 Detalhes de Implementação

A ferramenta ncRNA-Agentes foi criada conforme a arquitetura mostrada na Figura 4.1, e pode ser acessada em http://www.biomol.unb.br/ncrna-agents.

O sistema ncRNA-Agents foi implementado usando o Framework JADE versão 4.3.33, e o Drools 6.1.0 versão final para simular o raciocínio dos agentes.

Para a formalização do conhecimento biológico foram utilizado regras declarativas, de acordo com parâmetros adotados em cada projeto. Como exemplo, a Figura 4.2 mostra as regras da Camada de Resolução de Conflitos (CRC), que usam os resultados enviados pelos Agentes Gerentes da Camada Colaborativa.

As regras para o Gerente de Homologia foram criados para escolher, dentre os resultados dos três analistas (Blast, Infernal e tRNAscan), a melhor anotação, que é enviada para a CRC. O Analista tRNAscan verifica se existem bons alinhamentos, selecionando o alinhamento com maior *score*. O Analista Blast verifica se há algum alinhamento com $e - value \leq 10^{-5}$ (este limiar é um parâmetro facilmente alterável). O Analista Infernal verifica se há alinhamento com score ≥ 34 (outro parâmetro que pode ser modificado facilmente). A recomendação de anotação segue uma ordem escolhida pelo Gerente de Homologia, sendo a ordem adotada neste trabalho, Infernal, tRNAscan e Blast, para as ferramentas que encontrarem resultados, de acordo com os valores fixados para cada ferramenta.

Os Agentes Analistas executam em paralelo, usando *threads* para melhorar a eficiência. O Gerente Predição de Classe envia para a CRC a probabilidade da sequência investigada ser um ncRNA, calculado pelo analista Portrait, enquanto o Gerente *De novo* envia para a CRC o cálculo da energia mínima livre computada pelo Analista RNAfold. Os outros Analistas, que pertencem aos gerentes de Localização e Conservação Genômica, são usados como descrito na seção 4.3.2.

O servidor web foi implementado usando Apache Tomcat 7 e JSF versão 2.2.6.

4.4.1 Interface

No ncRNA-Agents, a interface com o usuário foi feita através de um projeto Web (Figura 4.3). A interface permite ao biólogo informar suas sequências no formato fasta, e definir os parâmetros de execução, tais como ferramentas e banco de dados de ncRNAs.

Depois da configuração feita pelo usuário, é necessário que os parâmetros selecionados sejam validados, e em caso de sucesso, essa requisição será submetida para o sistema.

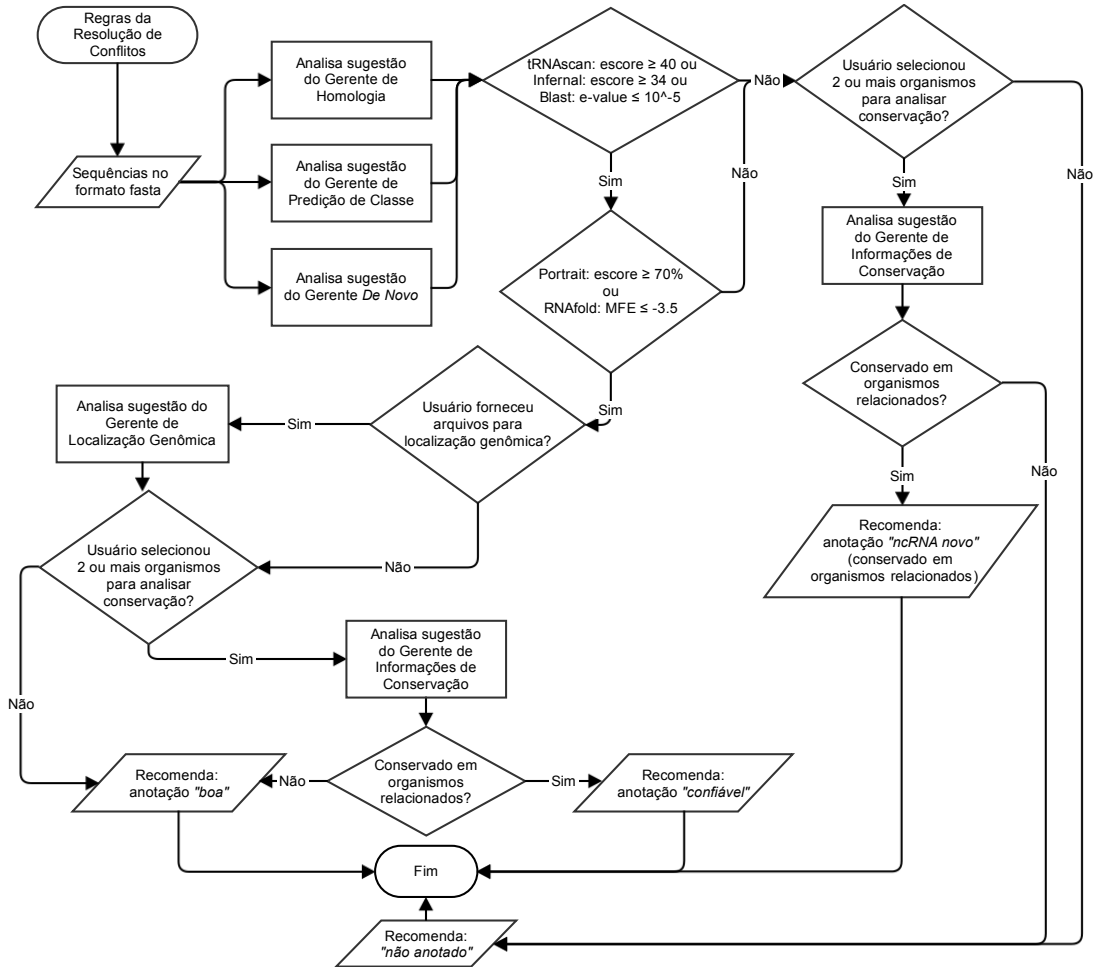


Figura 4.2: Regras utilizadas na Camada de Resolução de Conflitos, que simulam o raciocínio dos biólogos para analisar os resultados obtidos de ferramentas das três classes propostas na Seção 4.1 e na investigação entre organismos relacionados.

Caso a requisição tenha parâmetros inválidos, uma mensagem de erro será exibida para o usuário, destacando os campos onde foram encontrado erros.

Além da opção de submissão, a interface permite que o usuário acesse uma requisição já existente, na aba *Fetch request*, que tenha o no *status* de concluída ou em andamento. A aba *Case Study* fornece a opção de consultar estudos de caso já realizados.

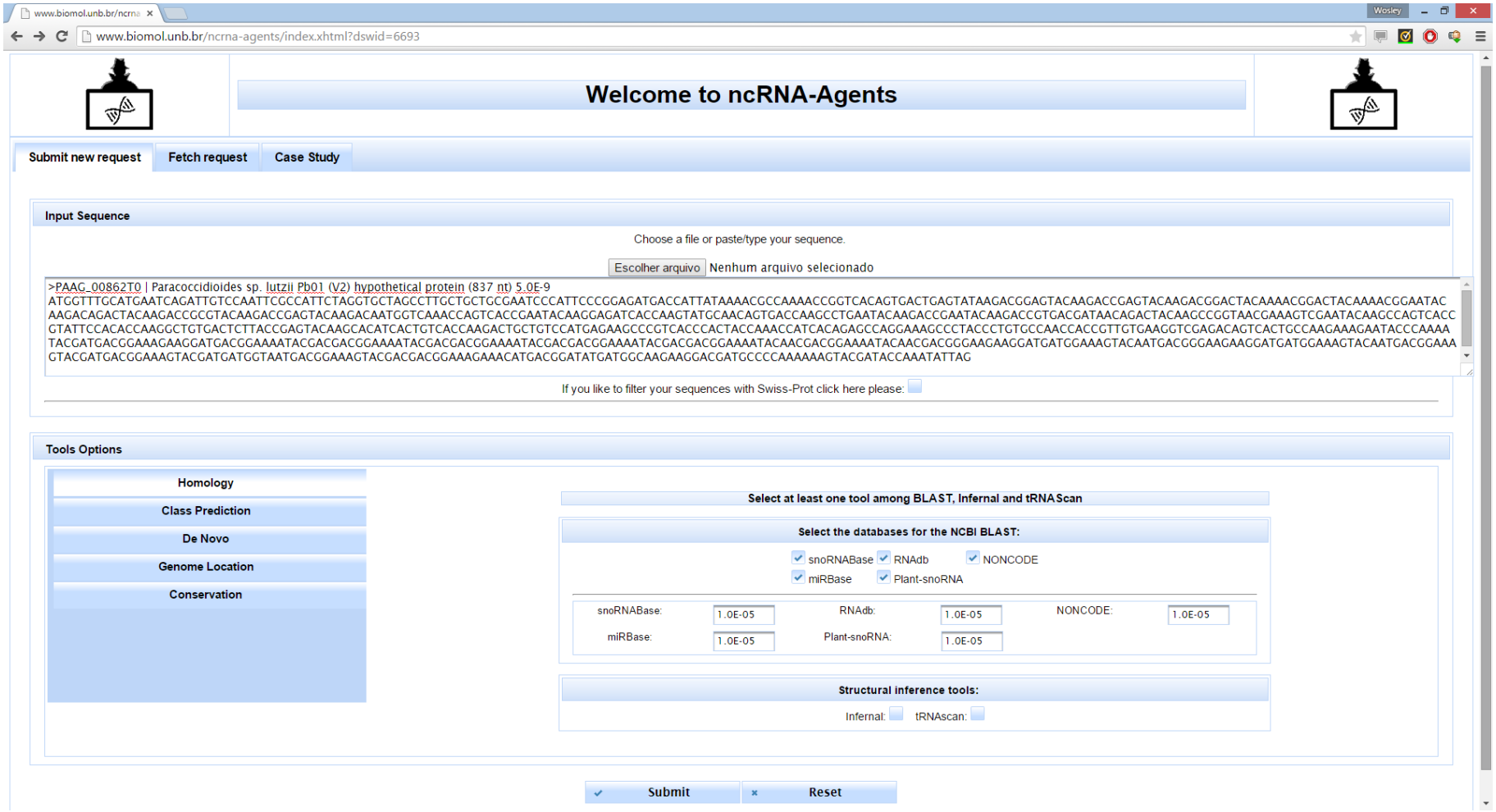


Figura 4.3: A interface Web do ncRNA-Agents.

Uma vez que a requisição é submetida, a interface espera pela primeira resposta do ncRNA-Agents que, ao ser obtida, redireciona o usuário para a página de resultados (Figura 4.4). Na página de resultados, uma mensagem informativa é apresentada ao usuário para lembrá-lo de guardar o identificador da requisição para posteriores consultas. A Figura 4.4 ilustra uma requisição com um único resultado para a sequência submetida. No caso de um arquivo com múltiplas sequências, vão ser apresentados ao usuário os resultados obtidos para todas as sequências. Se for necessário, os resultados podem ser mostrando em várias abas, conforme a quantidade de sequências submetidas.

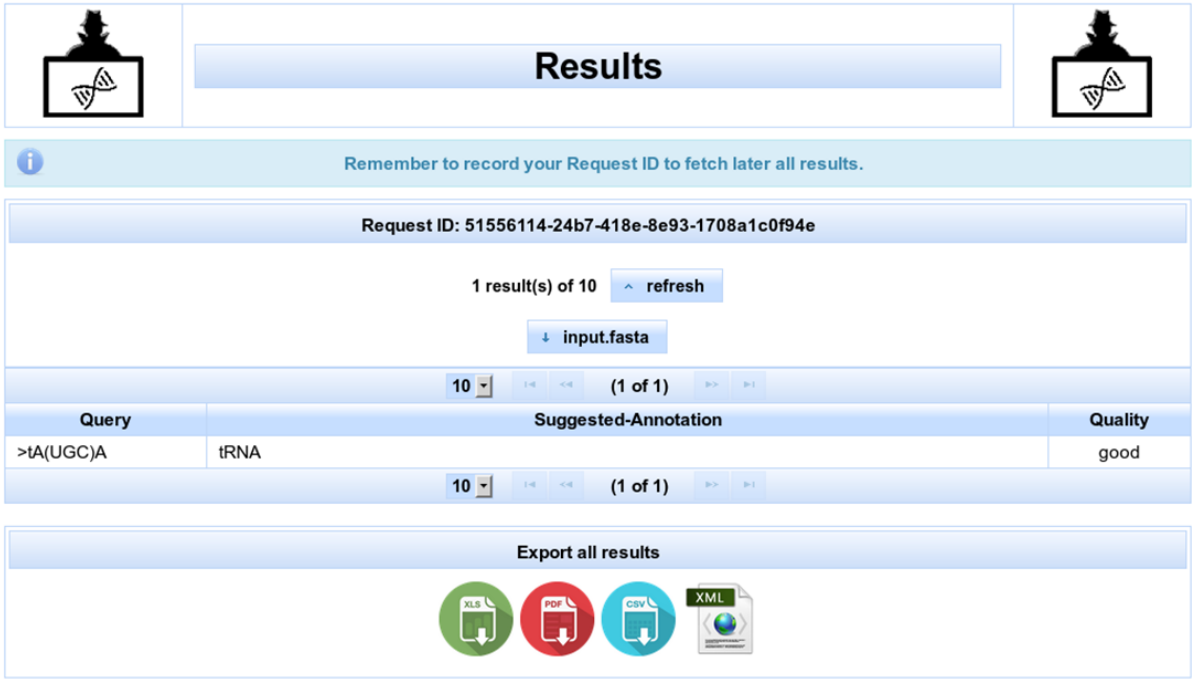


Figura 4.4: Página geral de resultados para uma sequência.

A primeira página tabela de resultados, apresentada na Figura 4.4, ilustra os resultados de maneira resumida. Para obter informações detalhadas de anotação de uma determinada sequência, o usuário necessita selecionar uma determinada linha de um desses resultados, como ilustrada na Figura 4.5. Nesta figura, são mostrados os resultados obtidos de cada ferramenta. Isso permite ao usuário a possibilidade de analisar de maneira detalhada os resultados e tirar suas próprias conclusões. Os resultados das anatações para as sequências podem ser exportados para os seguintes formatos: pdf, xml, xls e cvs.

As Figuras 4.6, 4.7, 4.8 e 4.9 mostram os resultados detalhados de cada ferramenta.

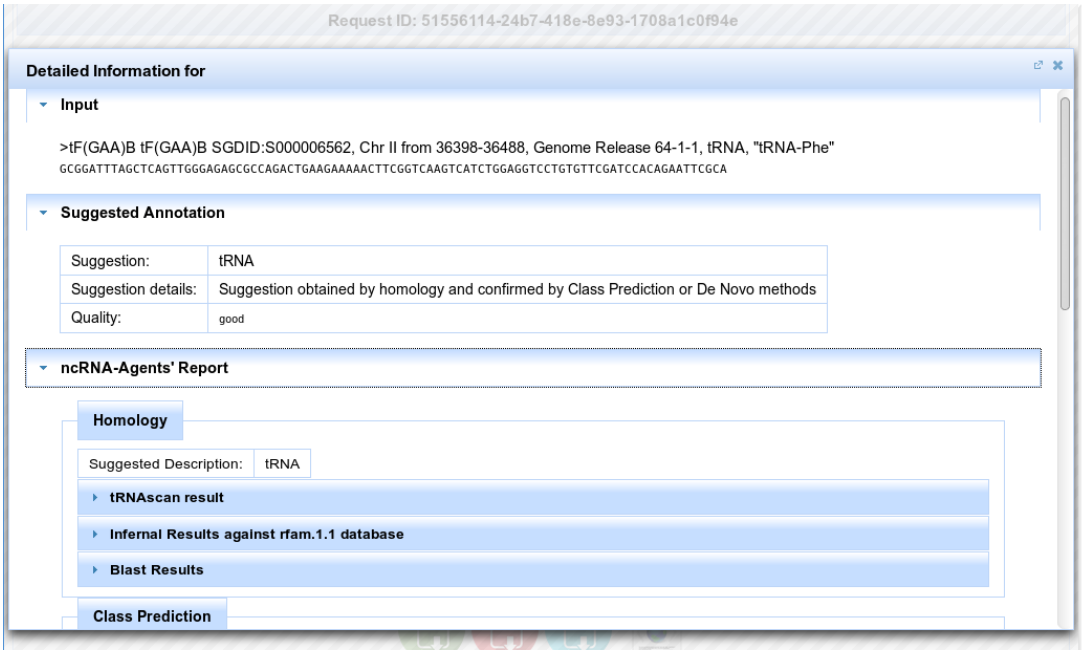


Figura 4.5: Página de resultados: anotação sugerida.

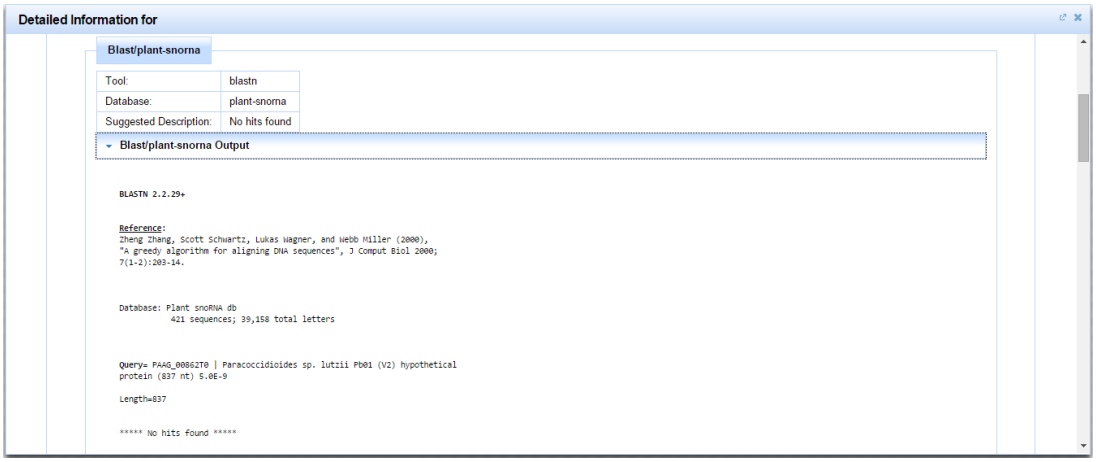


Figura 4.6: Resultados de Homologia, pelo Blast.

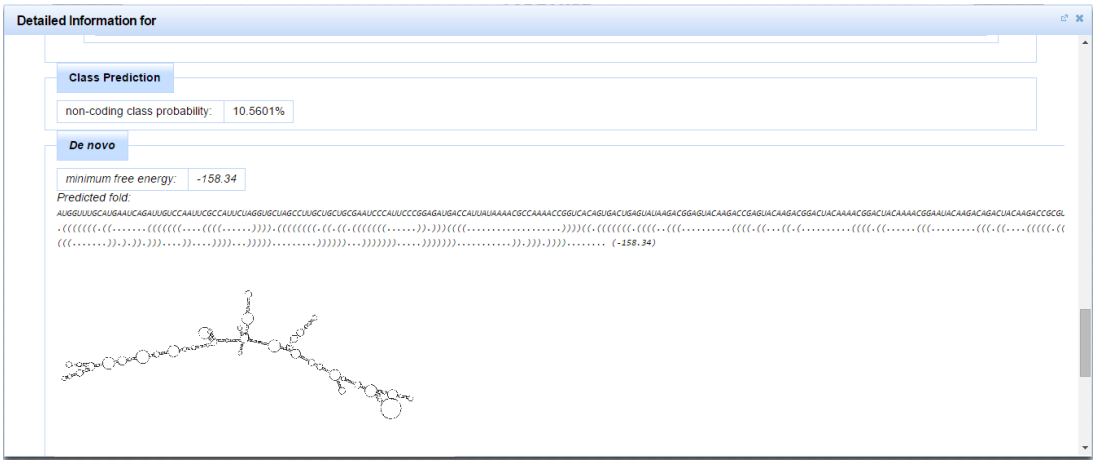


Figura 4.7: Resultados de Predição de Classe (SVM-Portrait) e a conformação espacial (gerada pelo RNAfold).

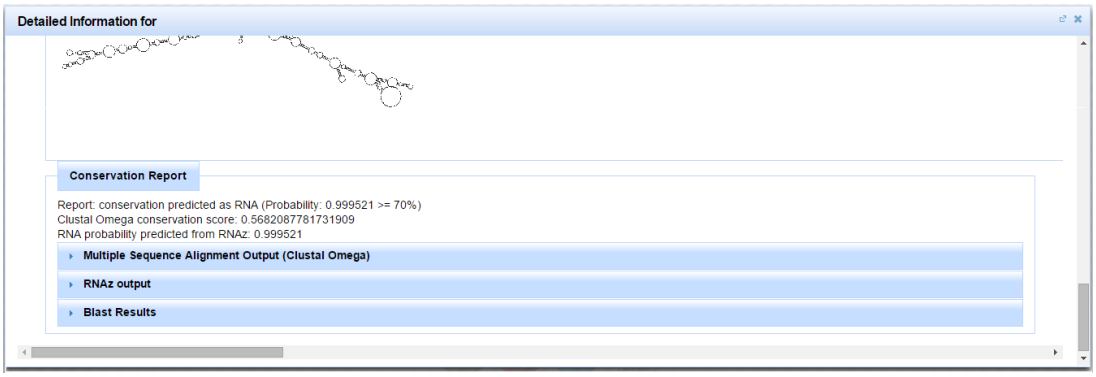


Figura 4.8: Resultados de Conservação entre organismos relacionados.

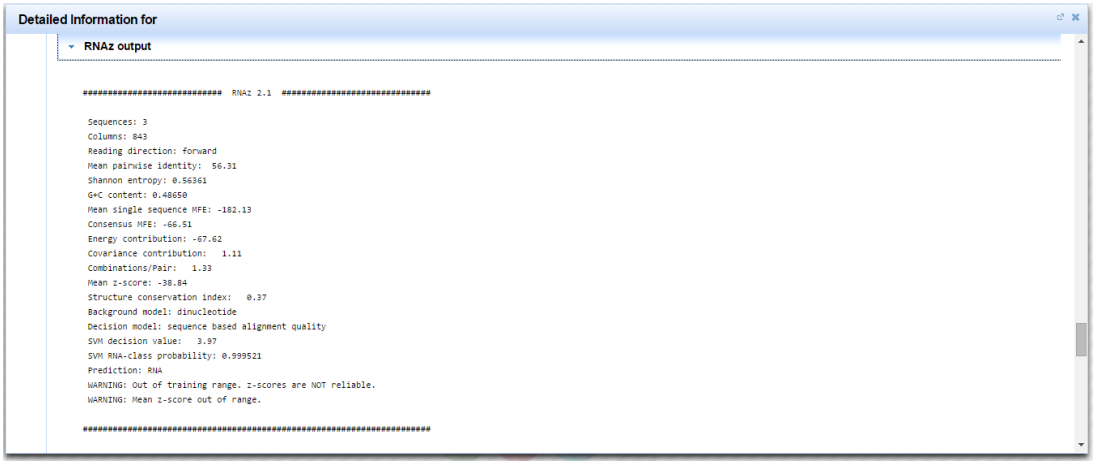


Figura 4.9: Resultados de Conservação da ferramenta gerado pelo RNAz.

Capítulo 5

Experimentos

Neste capítulo, discutimos três estudos de caso, realizados para avaliar o desempenho de predição de ncRNA do ncRNA-Agents. Primeiro na Seção 5.1, são comparadas anotações conhecidas de ncRNAs do fungo *Saccharomyces cerevisiae* com a anotação recomendado pelo ncRNA-Agentes, para medir o desempenho do sistema. Em seguida, investigamos ncRNAs dos fungos *Paracoccidioides brasilienses* e *Schizosaccharomyces pombe*, para descobrir novos ncRNAs, apresentadas nas Seções 5.2 e 5.3, respectivamente.

5.1 Avaliação de Performance

5.1.1 *Saccharomyces cerevisiae*

De acordo com a classificação taxonômica, a espécie *S. cerevisiae* (Figura 5.1) está incluída no Domínio Eukaryota; Reino Fungi; Filo Ascomycota; Classe Saccharomycetes; Ordem Saccharomycetales; Família Saccharomycetaceae e Gênero Saccharomyces [8]. O gênero Saccharomyces normalmente é formada por fungos unicelulares que se reproduzem assexuadamente por brotamento [14], ou sexuada por esporulação (meiose e segregação).

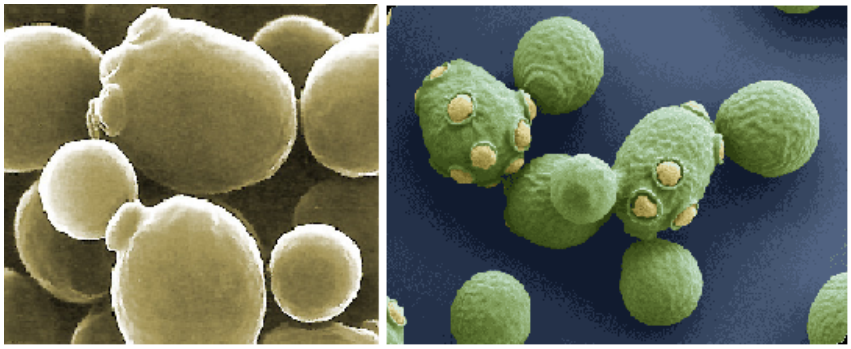


Figura 5.1: Saccharomyces: fungos unicelulares. Note que os pequenos brotos são novos indivíduos que estão sendo formados por reproduçãoo assexuada [14].

A *S. cerevisiae* é uma espécie de levedura utilizada há milhares de anos na fermentação de bebidas alcoólicas, tendo sido nos últimos anos utilizada na produção de bioetanol. Essa levedura é muito importante como organismo modelo [107], quer em estudos fisiológicos (etanol), ou na área da Biologia Celular e Molecular, sendo o microrganismo euca-

riótico mais estudado e cujo genoma foi o primeiro a ser sequenciado [153]. Isso justifica justifica o uso da *S. cerevisiae* como nosso padrão ouro.

5.1.2 Dados e Parâmetros Selecionados

Para medir o desempenho de predição de ncRNA do sistema ncRNA-Agentes, foi utilizado um conjunto (positivo) com 417 ncRNAs conhecidos, da *S. cerevisiae* s288C, compreendendo 77 snoRNAs, 299 tRNAs, 6 snRNAs, 1 RNA telomerase, 16 rRNAs e 18 predicted ncRNAs, disponível em http://www.yeastgenome.org/.

Para o conjunto negativo (proteínas), foram selecionados 417 RNAs codificadores de proteínas, transcritos da *S. cerevisiae* RM11-1a, disponíveis em http://www.broadinstitute .org/. Os transcritos da proteína foram escolhidos com base nos seguintes critérios: tamanho $\geq$ 30 nucleotídeos e $\leq$ 1.500 nucleotídeos; cada transcrito era alinhado com uma sequência armazenada no banco de dados SwissProt com e-value $\leq 10^{-10}$, com uma anotação funcional válida (diferente de "*hypothetical protein*" e termos relacionados); a anotação foi comparada com a anotação descrita de cada domínio da família, armazenada no banco de dados PRODOM, com uma pontuação $\geq 80\%$.

O sistema foi executado com todos os agentes mostrados na Figura 4.1. De acordo com o *Instituto Broad*, a divergência de sequência entre os conjuntos RM11-1a e S288C é estimado em 0.5% e 1%. Isso justifica o uso do conjunto correspondente RM11-1a como o genoma de referência, e o seu arquivo *.gtf* contendo o local de genes (Inicio, Fim das regiões, e exons), todos disponíveis em *http://www.broadinstitute.org/*. O arquivo *.gtf* foi usado para localizar os transcritos dos conjuntos positivos e negativos nas regiões intergênicas, exônicas ou intrônicas. Um local de transcrição é encontrado com base em um coeficiente de sobreposição (CS), de tal forma que se $CS \geq 80\%$ (o que significa que o transcrito tem uma sobreposição com uma região do genoma de pelo menos 80%), a localização da transcrição pode ser estimada no genoma de referência, o que permite identificar a sua região (íntron, éxon ou região intergênica).

5.1.3 Métricas para Avaliação dos Resultados

Para avaliar o desempenho do sistema ncRNA-Agents, em relação ao padrão ouro, foram utilizadas três métricas: acurácia, sensibilidade e especificidade [37, 155], tal que:

- Sensibilidade: expressa a capacidade de anotar corretamente ncRNAs

$$Sensibilidade = \frac{VP}{VP + FN} \tag{5.1}$$

- Especificidade: mede a capacidade de que um transcrito não predito como não codificador realmente não o seja

$$Especificidade = \frac{VN}{VN + FP} \tag{5.2}$$

- acurácia: mede a taxa geral de sucesso da ferramenta

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{5.3}$$

Para investigar os melhores parâmetros para tRNAscan e Portrait, foram realizados experimentos, mostrados na Figura 5.2. Esses experimentos indicam que a ferramenta Portrait não afeta a sensibilidade da ferramenta, ao contrário do tRNAscan, para qual o melhor *score* é de ≤ 30, que foi utilizado para todos os estudos de caso.

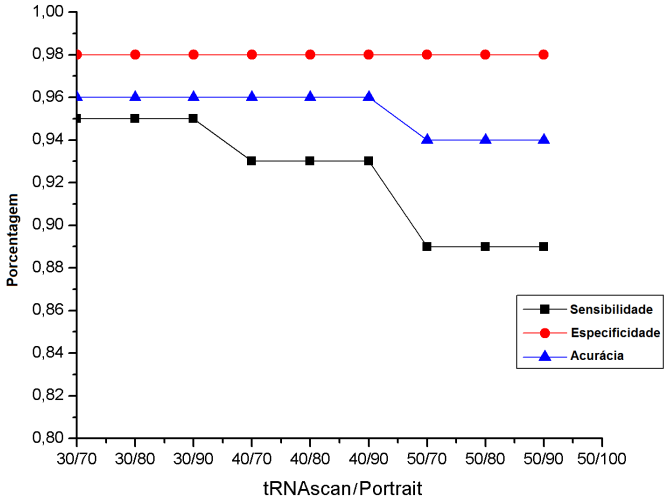


Figura 5.2: Sensibilidade, Especificidade e Acurácia para diferentes parâmetros de tRNAScan e Portrait.

A Tabela do Anexo A, mostra uma comparação entre a anotação feita pelo sistema ncRNA-Agents e Infernal, para todos os 417 ncRNAs (conjunto positivo) conhecidos. Nesta tabela, o ncRNA-Agents identificou 402 como ncRNAs (15 não foram anotadas como ncRNAs), enquanto o Infernal identificou 400 como ncRNAs (17 não foram anotadas como ncRNAs). Na verdade, ambos os métodos anotaram exatamente as mesmas funções, mas dois *mRNAlike lncRNAs* anotados pelo ncRNA-agentes não foram anotados pelo Infernal. Esses valores estão mostrados, na Tabela 5.1.

Tabela 5.1: Tabela de contingência (ncRNA-Agents)

Análise	Positivos ncRNAs	Negativos Codificadores de Proteínas
Sugestões positivas do ncRNA-Agents	402	24
Sugestões negativas do ncRNA-Agents	15	393

Em relação à 417 proteínas do conjunto negativo, o Infernal não anotou nenhum deles, enquanto o ncRNA-Agentes anotou 24 como ncRNAs novos (393 não foram preditos

como ncRNAs) os valores são mostrados na Tabela 5.2. Notamos que, desses 24 ncRNAs novos, 10 foram previamente anotados como RNAs ribossomais e duas proteínas como hipotéticas, sugerindo anotações diferentes das conhecidas. Isso mostra que o nosso sistema poderia ser usado para melhorar um grande número de anotações existentes, nos bancos de dados.

Tabela 5.2: Tabela de contingência (Infernal)

Análise	Positivos ncRNAs	Negativos Codificadores de Proteínas
Anotações positivas do Infernal	400	0
Anotações negativas do Infernal	17	417

Portanto, obtivemos as seguintes medidas para ncRNA-Agentes e Infernal, respectivamente: sensibilidade = 96.40% e 95.92%; especificidade = 94.24% e 100%; e acurácia = 95.32% e 97.96%.

5.2 NcRNAs Novos no *Paracoccidioides lutzii*

5.2.1 *Paracoccidioides brasiliensis*

Embora existam aproximadamente 100.000 espécies fúngicas, apenas uma pequena parcela apresenta risco para a saúde humana [124].

Dentro desse pequeno grupo de fungos patogênicos, estão os fungos dimórficos, que são responsáveis pela maior parte das infecções fúngicas sistêmicas de humanos [24, 103]. O processo da doença é marcado pela mudança de morfologia, onde o fungo transforma-se de uma forma infecciosa não patogênica, presente no ambiente, para uma forma patogênica, encontrada no tecido hospedeiro.

O *Paracoccidioides brasiliensis* (Figura 5.3) é uma causador de micose sistêmica humana, agente etiológico da *Paracoccidioidomicose* (PCM), uma doença restrita à América Latina, que foi descrita pela primeira vez por Adolph Lutz, em 1908, que isolou o fungo de pacientes. O *P. brasiliensis* tem o Domínio *eukaryoto*; Reino *Fungi*; Filo *Ascomycota*; Ordem *Onygenales*; Família *Onygenaceae* do qual também fazem parte os fungos *Histoplasma capsulatum*, *Coccidioides immitis*, *Blastomyces dermatitidis*, *Aspergillus nidulans* e *Aspergillus terreus* [103, 124, 125].

A PCM, antigamente chamada de blastomicose sul-americana, trata-se de uma micose sistêmica primária endêmica na região que se estende do México à Argentina, com maior incidência no Brasil (80% dos casos). O hospedeiro humano adquire o fungo por inalação de esporos da forma miceliana presente na natureza, que alcança os pulmões, e se converte na forma de levedura.

In vitro, e provavelmente na natureza, o fungo dimórfico *P. brasiliensis* é encontrado como micélio ou esporo à temperatura ambiente ou na forma de levedura. Quando o *P. brasiliensis* infecta o hospedeiro, ocorre o processo de transição dimórfica, provavelmente ativada pela mudança de temperatura, ocorrendo a conversão da forma miceliana para a forma de levedura. Essas observações sugerem fortemente que o processo dimórfico é

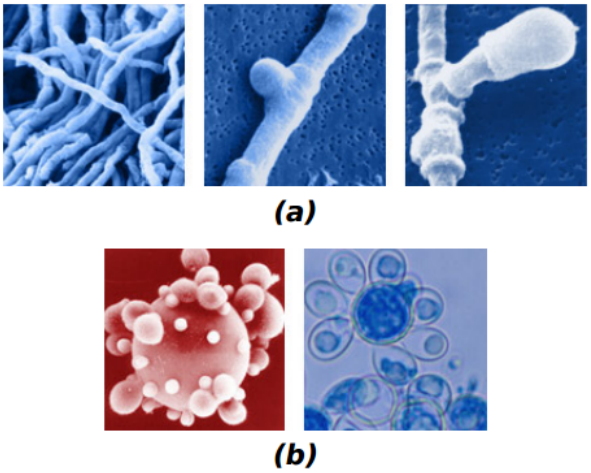


Figura 5.3: (a) *P. brasiliensis* em forma de micélio e (b) em forma de levedura [17].

um importante evento no estabelecimento da infecção, conforme observado para outros fungos patogênicos.

Em 2014 [139], mostraram que o *P. brasiliensis* Pb01 era uma espécie diferente do Pb18 e Pb03, por sua reprodução sexuada, entre outras características, denominando essa nova espécie de *P. lutzii*.

5.2.2 Dados e Parâmetros Selecionados

Os dados de 8,826 transcritos, de 110 supercontigos, do *P. lutzii* Pb01 foram obtidos em http://www.broadinstitute.org/.

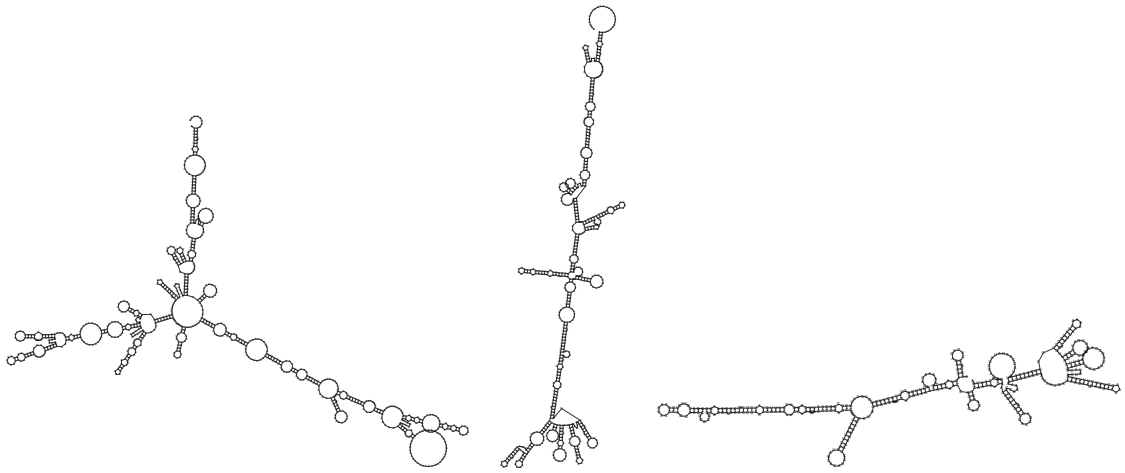
O mapeamento dos transcritos foi realizada com a ferramenta Segehmel [59], tomando como referência os supercontigs. Em seguida, um *script* Perl extraiu, com base no arquivo *.sam* gerado durante o mapeamento, a sequência de DNA do local mapeado nos supercontigs, para cada um dos 4,502 transcritos mapeados. Essas 4,502 sequências de DNA foram submetidos ao Blast com o banco de dados *SwissProt*, com e-value 10^{-5}, que anotou 2,517 proteínas. Em seguida as 1,985 proteínas que não foram anotadas, foram submetidas ao Blast com banco de dados UniPortKB, e 1,969 ("*no hits*") sequências não foram anotadas como proteínas. Ambo os conjuntos, de 16 sequências não anotadas e de 429 sequências anotada como "*hypothetical protein*" e termos relacionados (não só o *best hit*, mas todos os *hits* acima do limiar), perfazendo um total de 445 sequências, foram apresentados como entrada para ncRNA-Agents.

O sistema utilizou todos os gerentes e analistas mostrados na Figura 4.1. A Tabela 5.3 mostra os três ncRNAs novos anotados no *P. brasiliensis*. Os ncRNAs foram conservados em alguns fungos relacionados *Blastomyces dermatitidis* er 3.1, *Coccidioides immitis* rs finished, *Histoplasma capsulatum* h88, *Aspergillus fumigatus*, *Aspergillus nidulans* e *Aspergillus terreus*, todos eles obtidos em http://www.broadinstitute.org/. Os detalhes dessa anotação são mostrados na página do sistema http://www.biomol.unb.br/ncrna-agents.

Tabela 5.3: *Novel* ncRNAs em *P. brasiliensis* identificados por ncRNA-Agents.

Super-contig (v2)	Anotação ncRNA-Agents	Posição Inicial	Posição Final	Comprimento ncRNA (bp)	Fita	Legenda Figura
2	ncRNA novos	734814	733918	896	-	(a)
4	ncRNA novos	629991	630682	691	+	(b)
8	ncRNA novos	353736	354167	431	+	(c)

A Figura 5.4 mostra os ncRNAs identificados em *P. brasiliensis* pelo sistema ncRNA-Agents. Vale ressaltar que os ncRNAs previstos pelo ncRNA-agentes não têm quaisquer sequências comuns quando comparado com os preditos apenas por Portrait [6].



(a) SC 2 (b) SC 4 (c) SC 8

Figura 5.4: Estrutura espacial dos ncRNAs da Tabela 5.3, onde SC significa supercontig.

5.3 NcRNAs Novos no *Schizosaccharomyces pombe*

5.3.1 *S. pombe*

Schizosaccharomyces pombe, também chamada "levedura de fissão", é usada como organismo modelo em Biologia Molecular e Celular. Trata-se de um eucarioto unicelular, cujas células têm forma de bastão, medindo tipicamente entre 3 e 4 micrómetros de diâmetro e 7 a 14 micrômetros de comprimento. Estima-se que o seu genoma tenha cerca de 14,1 milhões de pares de bases.

As células mantêm a sua forma crescendo exclusivamente a partir das extremidades e dividindo-se por fissão para produzirem duas células-filhas de tamanhos iguais, o que as torna uma ferramenta poderosa no estudo do ciclo celular.

A sequência do genoma de *S. pombe* foi publicada em 2002, por um consórcio liderado pelo Wellcome Trust Sanger Institute, tornando-se o sexto organismo modelo eucariota cujo genoma foi totalmente sequenciado [156].

Em 2006, foi publicada a localização de todas as proteínas em *S. pombe*, feita utilizando proteína verde fluorescente como marcador molecular [87].

De acordo com a classificação taxonômica, a espécie *S. pombe* (Figura 5.5) está incluída no Domínio Eukaryota; Reino Fungi; Filo Ascomycota; Classe Schizosaccharomycetes; Ordem Schizosaccharomycetales; Família Schizosaccharomycetaceae e Gênero Schizosaccharomyces.

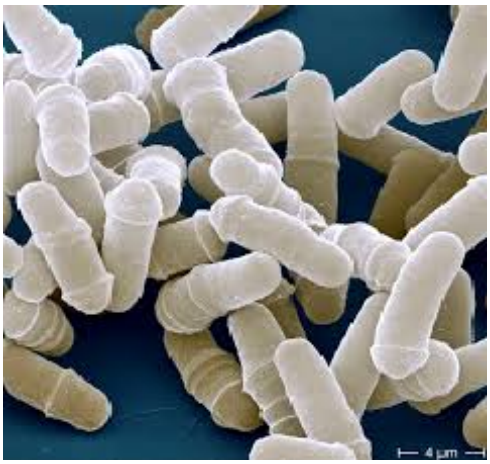


Figura 5.5: A *Schizosaccharomyces pombe*, é também chamada de"levedura de fissão" [17].

5.3.2 Dados e Parâmetros Selecionados

Um outro estudo de caso foi desenvolvido com o *S. pombe*, no qual foram utilizadas informações do genoma e do transcritoma desse fungo.

Dados do RNA-Seq do *S. pombe* foram extraídos do EMBL-EBI (experimento E-MTAB-1154). Inicialmente, foi feita uma filtragem para remover as sequências com baixa qualidade, e em seguida foi realizado um mapeamento com a ferramenta Segehmelh [59], tomando como referência o genoma *S. pombe* (obtidos de http://www.broadinstitute.org/). Em seguida, um *script* Perl foi utilizado para extrair as regiões obtidas no mapeamento que apresentaram no mínimo 5 sequências de cobertura. Depois, outro *script* Perl buscou a localização de cada região no arquivo *.sam* com as sequências geradas do genoma da *S. pombe*. A regiões mapeadas foram submetidos ao Blast com banco de dados Swiss-Prot. Sequências não identificadas como proteínas, ("no hits found"), juntamente com sequências anotadas como "*hypothetical protein*", "*predicted protein*", "*potential protein*" e "*unknown sequence*", foram submetidos como entrada para ncRNA-Agents.

Como nos dois experimentos anteriores, todos os agentes da Figura 4.1 foram utilizados. Na localização genômica, o Agente Analista de Localização Genômica utilizou o arquivo *.gtf* para localizar se a sequência ocorria em éxon, íntron ou região intergênica. Uma sequência foi considerado como localizada em uma região intrônica se apresentasse uma intersecção de, no máximo, 70% com um íntron.

Nos três cromossomos do fungo *S. pombe* foram encontradas $9,566$, $7,565$ e $4,126$ regiões respectivamente. Para cada uma dessas regiões, um Blast contra SwissProt foi executado e $1,795$, $1,562$ e 802 sequências, nos cromossomos I, II e III, foram anotadas

como proteínas. Assim, um total de 17,098 sequências (7,771 do cromossomo I, 6,003 do cromossomo II e 3,324 do cromossomo III) foram submetidas como entrada para o ncRNA-Agentes, que detectou 40 novos ncRNAs (21 como conservação e 19 sem conservação em organismos relacionados), todos na "fita +", tendo sido ambas fitas verificadas. As características gerais destes ncRNAs são mostrados nas Tabelas 5.4 e 5.5, com estruturas espaciais mostradas nas Figuras 5.6 e 5.7.

Tabela 5.4: NcRNAs no *S. pombe* identificados pelo ncRNA-agents nos cromossomos I, II e III apresentando conservação em organismos relacionados, todos encontrados na fita +.

Cromossomo	Anotação ncRNA-Agents	Posição Inicial	Posição Final	Comprimento ncRNA (bp)	Legenda Figura
I	tRNA	365148	365236	88	(a)
I	snoRNA	1772288	1772406	118	(b)
I	snoRNA	2441955	2442050	95	(c)
I	tRNA	3086302	3086405	103	(d)
I	snoRNA	3700659	3700749	90	(e)
I	tRNA	3701548	3701638	90	(f)
I	snoRNA	3776767	3776848	81	(g)
II	tRNA	355913	356005	92	(h)
II	rRNA	570150	570239	89	(i)
II	tRNA	599696	1599799	103	(j)
II	tRNA	1645034	1645137	103	(k)
II	snRNA	3236865	3236992	127	(l)
II	snRNA	3350583	3350694	111	(m)
II	tRNA	3814123	3814225	102	(n)
II	tRNA	3939221	3939306	85	(o)
III	snR99	583418	583614	196	(p)
III	tRNA	847694	847787	93	(q)
III	tRNA	925843	925933	93	(r)
III	tRNA	1071041	1071190	149	(s)
III	tRNA	1102800	1102922	122	(t)
III	tRNA	1863119	1863215	96	(u)

Note que, em ambos os casos (os ncRNAs preditos foram obtidos a partir de todos os gerentes e analistas), para ser previsto como ncRNAs novos, a sequência precisa apresentar a energia livre de dobramento < -3.50 kcal/mol (de acordo com RNAfold e RNAz), o que mostrava as estruturas conservadas estáveis.

A conservação para as sequências do fungo *S. pombe* foi investigada nos seguintes organismos (os organismos mais próximos com informação disponível): *Pneumocystis murina*, *Schizosaccharomyces cryophilus*, *Schizosaccharomyces japonicus* e *Schizosaccharomyces Octosporus*, todos eles obtidos em http://www.broadinstitute.org/.

Sequências não anotadas como ncRNAs pelos Gerente de Homologia, Predição de Classe ou *De novo* foram submetidos ao Gerente de Localização Genômica (usando anotação do arquivo .gtf do *S. pombe*) para o Gerente Conservação, que permitiu encontrar:

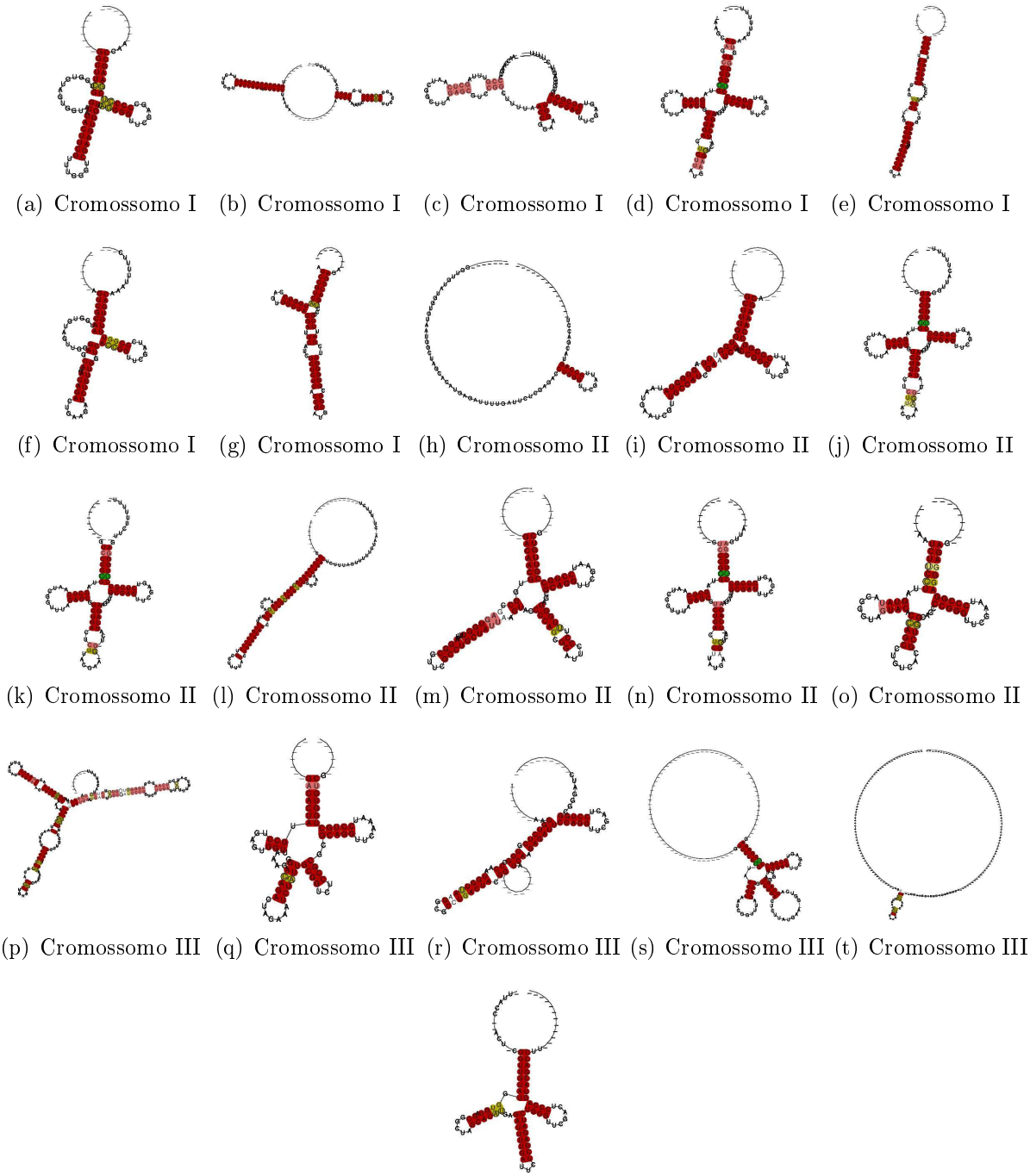
- 21 ncRNAs novos, todos eles apresentando conservação em, pelo menos, dois organismos relacionados: (i) no cromossomo I, 6 em exons (3 tRNAs, 3 snoRNA) e 1 snoRNA em intron; (ii) e cromossomo II, 8 em exons (5 tRNAs, 1 rRNA, 2 snRNA); e (iii) no cromossomo III, 6 em exons (5 tRNAs, 1 snRNA);

Tabela 5.5: NcRNAs identificados na *S. pombe* pelo ncRNA-Agents nos cromossomos I, II e III, não apresentando conservação em organismos relacionados, todos encontrados na fita +.

Cromossomo	Anotação ncRNA-Agents	Posição Inicial	Posição Final	Comprimento ncRNA (bp)	Legenda Figura
I	snoRNA	431217	431301	84	(a)
I	snoRNA	526920	527092	172	(b)
I	snoRNA	527188	527301	113	(c)
I	snoRNA	1200132	1200257	125	(d)
I	snoRNA	1200391	1200508	117	(e)
I	snoRNA	1712992	1713099	107	(f)
I	snoRNA	2416362	2416500	138	(g)
I	snoRNA	3699719	3699850	131	(h)
I	snoRNA	4516055	4516155	100	(i)
I	mRNAlike	4516842	4516892	50	(j)
I	mRNAlike	5021770	5021853	83	(k)
II	mRNAlike	1965085	1965168	83	(l)
II	mRNAlike	1969651	1969734	83	(m)
II	snoRNA	2044199	2044327	128	(n)
II	snRNA	3020253	3020353	100	(o)
II	tRNA	3814551	3814659	108	(p)
II	snoRNA	3944593	3944692	99	(q)
III	snRNA	2419266	2419316	50	(r)
III	snRNA	2419595	2419645	50	(s)

- 19 ncRNAs novos, sem conservação em organismos relacionados: (i) no cromossomo I, 9 em exons (8 snoRNAs, 1 lncRNA), e 1 lncRNA, 1 snRNA em intron; (ii) no cromossomo II, 4 em exons (1 tRNA, 1 snRNA, 2 snoRNAs) e 2 lncRNAs em introns; e (iii) no cromossomo III, 2 snRNAs em exons.

A Tabela 5.6 mostra o número total e tipos de ncRNAs identificados no *S. pombe*, com/sem conservação em organismos relacionados.



(a) Cromossomo I (b) Cromossomo I (c) Cromossomo I (d) Cromossomo I (e) Cromossomo I

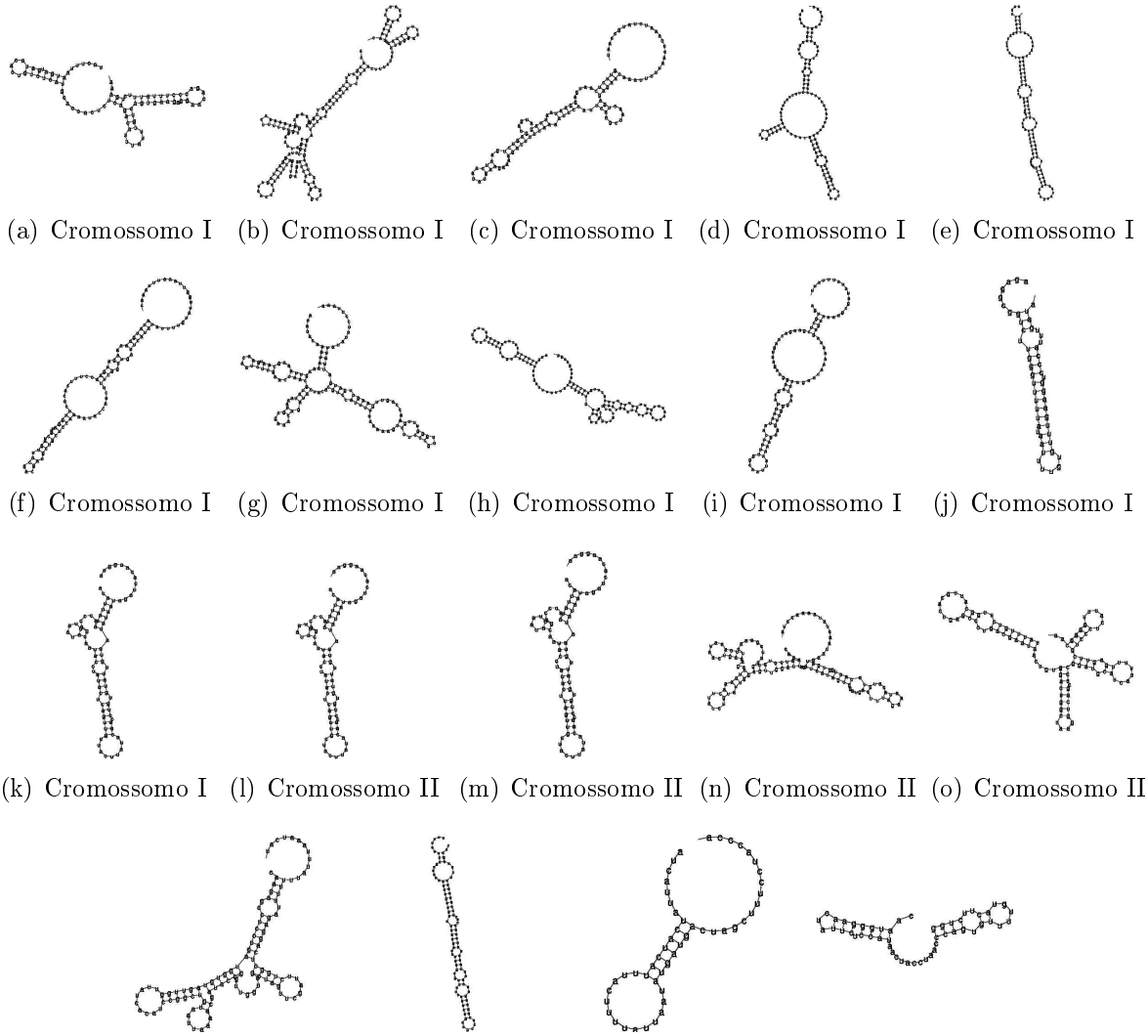
(f) Cromossomo I (g) Cromossomo I (h) Cromossomo II (i) Cromossomo II (j) Cromossomo II

(k) Cromossomo II (l) Cromossomo II (m) Cromossomo II (n) Cromossomo II (o) Cromossomo II

(p) Cromossomo III (q) Cromossomo III (r) Cromossomo III (s) Cromossomo III (t) Cromossomo III

(u) Cromossomo III

Figura 5.6: Estruturas espaciais dos ncRNAs da Tabela 5.4, conservada em, pelo menos, dois fungos relacionados.



(a) Cromossomo I (b) Cromossomo I (c) Cromossomo I (d) Cromossomo I (e) Cromossomo I

(f) Cromossomo I (g) Cromossomo I (h) Cromossomo I (i) Cromossomo I (j) Cromossomo I

(k) Cromossomo I (l) Cromossomo II (m) Cromossomo II (n) Cromossomo II (o) Cromossomo II

(p) Cromossomo II (q) Cromossomo II (r) Cromossomo III (s) Cromossomo III

Figura 5.7: Estruturas espaciais dos ncRNAs da Tabela 5.5, não conservadas em fungos relacionados.

Tabela 5.6: Total de ncRNAs do *S. pombe*, com/sem conservação em fungos relacionados.

	Tipo					
Cromossomo	tRNA exon	ncRNA exon	rRNA exon	snRNA exon	intron	exon
I	2/0	3/7	0/0	0/0	1/2	1/2
II	2/1	1/1	2/0	1/2	0/2	2/0
III	2/0	1/2	0/0	0/0	0/0	3/0
Total	6/1	5/10	2/0	1/2	1/4	6/2

Capítulo 6

Conclusão

Nesta tese, foi apresentado o ncRNA-agentes, um sistema multiagente para recomendar anotação de ncRNA. A arquitetura proposta permite a execução de agentes inteligentes que colaboraram em um ambiente heterogêneo e dinâmico. Diferentes ferramentas e banco de dados, bem como regras de inferência que simulam o conhecimento dos biólogos, podem ser especificado de acordo com os objetivos de cada projeto. No ncRNA-agentes, os agentes são especializados em diferentes tarefas, e podem agir de forma independente usando regras de inferência distintas nos agentes.

Um sistema foi implementado utilizando o *framework* JADE para o SMA, *Drools* como motor de inferência de regras de raciocínio, em um ambiente web. Para validar o sistema. Foi realizado um estudo de caso para detectar ncRNAs na *Saccharomyces cerevisiae*, sendo esses resultados comparados aqueles obtidos com a ferramenta Infernal. Os ncRNAs anotados na *S. cerevisiae* foram considerados o padrão-ouro do nosso estudo. Além disso, fizemos mais dois outros experimentos para detectar ncRNAs nos fungos *Schizosaccharomyces pombe* e *Paracoccidioides brasiliensis*. Para ambos, obtivemos ncRNAs novos, procurando por conservação em organismos relacionados.

Acreditamos que o ncRNA-agents ajuda a melhorar a qualidade da anotação de ncRNAs, considerando que os sequenciadores de alto desempenho produzem milhões de pequenos fragmentos. Tanto quanto sabemos, este sistema é único, tanto em relação à simulação de raciocínio dos biólogos, quanto à possibilidade de integração de diferentes ferramentas e bancos de dados, utilizando uma abordagem multiagente.

6.1 Contribuições

Em seguida, as contribuições mais importantes desta tese são apresentada:

- Arquitetura baseada na execução de agentes inteligentes cooperativa em um ambiente observável, probabilístico, episódico, dinâmico, discreto e multiagente, incluindo diferentes ferramentas e bancos de dados, e regras de inferência que simulam o raciocínio dos biólogos. O sistema pode ser configurado de acordo com os objetivos de cada projeto;
- Um sistema foi implementado usando JADE e Drools em um ambiente web, disponível publicamente em http://www.biomol.unb.br/ncrna-agents/index;

- Três experimentos com os fungos *S. cerevisiae*, *P. brasilienses* e *S. pombe* apontaram ncRNAs novos, que também estão disponíveis na página.

Os artigos aceitos foram:

- ncRNA-Agents: a multiagent system for non-coding RNA annotation - Ver Anexo B
- Knowledge based reasoning to annotate non-coding RNA using multi-agent system - Ver Anexo C

6.2 Trabalhos Futuros

Os próximos passos incluem melhorar os conhecimentos dos agentes em ncRNA-agentes, usando outros mecanismos de raciocínio. Particularmente, na Camada Colaborativa, outros agentes analistas (correspondentes a outras ferramentas) poderiam facilmente ser incluídos por exemplo, snoReport [54] e o snoStrip [9]. No Gerente de Predição Classe, uma ferramenta para classificar ncRNAs poderia ser incluída, além do PORTRAIT, que apenas indica se uma sequência é um potencial ncRNA. Métodos de mineração de dados podem ser usados para ampliar o número e variedade de organismos na camada física, bem como para descobrir novos padrões de ncRNAs, além de disponibilizar anotações de ncRNAs para diferentes organismos no sistema ncRNA-Agents.

Além disso, aspectos de implementação devem ser melhorados permitindo uma ferramenta reprodutível e portátil que possa ser integrada a sistemas de anotação diferentes. Uma aplicação distribuída poderia ser usada para reduzir o tempo de execução, permitindo utilizar ncRNA-Agentes em projetos de sequenciamento de alto desempenho.

Referências

[1] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, D.J. Lipman, et al. Basic Local Alignment Search Tool. *Journal of Molecular Biology*, 215(3):403–410, 1990. 18, 19, 21, 51, 55

[2] J. M. Amabis and G. R. Martho. *Fundamentos da Biologia Moderna*. Moderna, 1996. 6

[3] P. P. Amaral, M. E. Dinger, T. R. Mercer, and J. S. Mattick. The Eukaryotic Genome as an RNA Machine. *Science*, 319(5871):1787–1789, 2008. 12

[4] M. Aparicio, L. Chiariglione, E. Mamdani, F. McCabe, R. Nicol, D. Steiner, and H. Suguri. FIPA-intelligent Agents from Theory to Practice. In *GLORI 2nd Conf. On Dynamic Logistics*, 1999. 34

[5] R. T. Arrial. Predição de RNAs não-codificadores no transcriptoma do fungo *Paracoccidioides brasiliensis* usando aprendizagem de máquina. *Dissertação de Mestrado do Instituto de Biologia UnB*, 2006. 20, 21, 52, 55

[6] R.T. Arrial, R.C. Togawa, and M.M. Brigido. Screening non-coding RNAs in transcriptomes from neglected species using PORTRAIT: case study of the pathogenic fungus Paracoccidioides brasiliensis. *BMC Bioinformatics*, 10(1):239, 2009. 20, 52, 67

[7] J.H. Badger and G.J. Olsen. CRITICA: coding region identification tool invoking comparative analysis. *Molecular Biology and Evolution*, 16(4):512–524, 1999. 12, 18

[8] J.A. Barnett. The taxonomy of the genus Saccharomyces meyen ex Reess: a short review for non-taxonomists. *Yeast*, 8(1):1–23, 1992. 62

[9] S. Bartschat, S. Kehr, H. Tafer, P. F. Stadler, and J. Hertel. snoStrip: A snoRNA annotation pipeline. *Bioinformatics*, page btt604, 2013. 74

[10] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa. JADE - a white paper. *White Paper 3, TILAB - Telecom Italia Lab. (http://jade.tilab.com/)*, v. 3:p. 6–19, 2003. 43

[11] F. L. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-agent Systems with JADE*, volume 7. John Wiley & Sons, 2007. 37, 38

[12] E. Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America*, 99(Suppl 3):7280–7287, 2002. 25

[13] R. H. Bordini, J. F. Hübner, and M. Wooldridge. *Programming Multi-agent Systems in AgentSpeak using Jason*, volume 8. John Wiley & Sons, 2007. 43

[14] C. Boulton and D. Quain. *Brewing Yeast and Fermentation*. John Wiley & Sons, 2008. 62

[15] M. Bratman. Intention, plans, and practical reason. *Harvard University*, 1987. 40

[16] J. Brennecke, A.A. Aravin, A. Stark, M. Dus, M. Kellis, R. Sachidanandam, and G.J. Hannon. Discrete small RNA-generating loci as master regulators of transposon activity in Drosophila. *Cell*, 128(6):1089–1103, 2007. 13, 16, 17

[17] Broad, 2015. http://www.broadinstitute.org/. 66, 68

[18] R. A. Brooks. Intelligence without reason, computers and thought lecture. *Proceedings of IJCAI-91, Sidney, Australia*, 1991. 32, 37, 38, 39

[19] J.W.S. Brown, M. Echeverria, L.H. Qu, T.M. Lowe, J.P. Bachellerie, A. Hüttenhofer, J.P. Kastenmayer, P.J. Green, P. Shaw, and D.F. Marshall. Plant snoRNA database. *Nucleic Acids Research*, 31(1):432–435, 2003. 22

[20] T. A. Brown. *Genomes*. Wiley-Liss, Oxford, 2006. 12, 15

[21] P. Browne. *JBoss Drools Business Rules*. Packt Publishing, 2009. 46, 47

[22] S. W. Burge, J. Daub, R. Eberhardt, J. Tate, L. Barquist, E. P. Nawrocki, S. R. Eddy, P. P. Gardner, and A. Bateman. Rfam 11.0: 10 years of RNA families. *Nucleic Acids Research*, page gks1005, 2012. 15, 22

[23] DA SILVA JÚNIOR CÉSAR and SEZAR SASSON. Biologia-volume único 4ª edição. *Editora: Saraiva*, 2007. 7

[24] D. Chen, T. K. Janganan, G. Chen, E. R. Marques, M. R. Kress, G. H. Goldman, A. R. Walmsley, and M. I. Borges-Walmsley. The camp pathway is important for controlling the morphological switch to the pathogenic yeast form of paracoccidioides brasiliensis. *Molecular Microbiology*, 65(3):761–779, 2007. 65

[25] F. Christodoulou, F. Raible, R. Tomer, O. Simakov, K. Trachana, S. Klaus, H. Snyman, G. J. Hannon, P. Bork, and D. Arendt. Ancient animal microRNAs and the evolution of tissue identity. *Nature*, 463(7284):1084–1088, 2010. 13

[26] C. Chu, K. Qu, F. L. Zhong, S. E. Artandi, and H. Y. Chang. Genomic maps of long noncoding RNA occupancy reveal principles of RNA-chromatin interactions. *Molecular Cell*, 44(4):667–678, 2011. 12

[27] G. Condorelli and S. Dimmeler. MicroRNAs: components of an integrated system controlling cardiac development, physiology, and disease pathogenesis. *Cardiovascular Research*, 2008. 13

[28] F. F. Costa. Non-coding RNAs: Meet thy masters. *Bioessays*, 32(7):599–608, 2010. 17

[29] X. Darzacq, B. E. Jády, C. Verheggen, A. M. Kiss, E. Bertrand, and T. Kiss. Cajal body-specific small nuclear RNAs: a novel class of 2' O-methylation and pseudouridylation guide RNAs. *The EMBO Journal*, 21(11):2746–2756, 2002. 16, 17

[30] S. Deryusheva and J. G. Gall. Novel small Cajal-body-specific RNAs identified in Drosophila: probing guide RNA function. *RNA*, 19(12):1802–1814, 2013. 16, 17

[31] J. Dopp and J. Austin. How to do things with words. *Revue Philosophique de Louvain*, 60(68):704–705, 1962. 33

[32] R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998. 13, 16, 17, 55

[33] S. R. Eddy. Computacional genomics of noncoding RNA genes. *Cell*, 109:137–140, 2002. 18

[34] S. R. Eddy. Infernal user's guide. *http://infernal.janelia.org*, 2003. 19, 51, 55

[35] S.R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Research*, 22(11):2079–2088, 1994. 19, 20, 21

[36] S.R. Eddy et al. Non-coding RNA genes and the modern RNA world. *Nature Reviews Genetics*, 2(12):919–929, 2001. x, 11, 13, 16, 17, 23

[37] T. Fawcett. Roc graphs: Notes and practical considerations for researchers. *Machine Learning*, 31:1–38, 2004. 63

[38] J. Ferber. *Multi-agent Systems: an Introduction to Distributed Artificial Intelligence*, volume 1. Addison-Wesley Reading, 1999. 29

[39] G.A. Fichant and C. Burks. Identifying potential tRNA genes in genomic DNA sequences. *Journal of Molecular Biology*, 220(3):659, 1991. 20

[40] A. Fip. FIPA Contract Net Interaction Protocol Specification, 2001. 37

[41] ACL FIPA. FIPA ACL Message Structure Specification. *Foundation for Intelligent Physical Agents, http://www.fipa.org/specs/fipa00061/SC00061G.html (30.6. 2004)*, 2002. 34, 35

[42] C.L. Forgy. Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence*, 19(1):17–37, 1982. 46, 47

[43] E. Friedman-Hill. *JESS in Action*. Manning, 2003. 46

[44] V. K. Gangaraju and H. Lin. MicroRNAs: key regulators of stem cells. *Nature Reviews Molecular Cell Biology*, 10(2):116–125, 2009. 13

[45] Genbank, 2012. http://www.ncbi.nlm.nih.gov/Genbank/. 21

[46] E. A. Gibb, C. J. Brown, and W. L. Lam. The functional role of long non-coding RNA in human carcinomas. *Mol Cancer*, 10(1):38–55, 2011. 17

[47] W. Giffiths and G. Lewontin. Introdução à genética, 8º edição, 2006. 8

[48] S. Griffiths-Jones. Annotating noncoding RNA genes. *Annu. Rev. Genomics Hum. Genet.*, 8:279–298, 2007. 23, 51

[49] S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna, and S.R. Eddy. Rfam: an RNA family database. *Nucleic Acids Research*, 31(1):439–441, 2003. 15, 22

[50] S. Griffiths-Jones, R. J. Grocock, S. Dongen, A. Bateman, and A. J. Enright. miRBase: microRNA sequences, targets and gene nomenclature. *Nucleic Acids Research*, 34:D140–D144, 2006. 22

[51] A.R. Gruber, R. Neuböck, I.L. Hofacker, and S. Washietl. The RNAz web server: prediction of thermodynamically stable and evolutionarily conserved RNA structures. *Nucleic Acids Research*, 35(suppl 2):W335–W338, 2007. 20, 21

[52] Hammurapi, 2012. http://www.hammurapi.biz/hammurapi-biz/ef/xmenu/hammurapi-group/products/hammurapi-rules/. 46, 47

[53] S. Henikoff and J.G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992. 19

[54] J. Hertel, I. L. Hofacker, and P. F. Stadler. SnoReport: computational identification of snoRNAs with unknown targets. *Bioinformatics*, 24(2):158–164, 2008. 74

[55] S. Hirotsune, N. Yoshida, A. Chen, L. Garrett, F. Sugiyama, S. Takahashi, K. Yagami, A. Wynshaw-Boris, and A. Yoshiki. An expressed pseudogene regulates the messenger-RNA stability of its homologous coding gene. *Nature*, 423(6935):91–96, 2003. 17

[56] I.L. Hofacker. Vienna RNA secondary structure server. *Nucleic Acids Research*, 31(13):3429–3431, 2003. 20

[57] I.L. Hofacker, M. Fekete, and P.F. Stadler. Secondary structure prediction for aligned RNA sequences. *Journal of Molecular Biology*, 319(5):1059–1066, 2002. 23

[58] I.L. Hofacker, W. Fontana, P.F. Stadler, L.S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie/Chemical Monthly*, 125(2):167–188, 1994. 20, 21, 23

[59] S. Hoffmann, C. Otto, S. Kurtz, C. M. Sharma, P. Khaitovich, J. Vogel, P. F. Stadler, and J. Hackermüller. Fast mapping of short sequences with mismatches, insertions and deletions using index structures. *PLoS computational biology*, 5(9):e1000502, 2009. 66, 68

[60] M. Huarte, M. Guttman, D. Feldser, M. Garber, M. J. Koziol, D. Kenzelmann-Broz, A. M. Khalil, O. Zuk, I. Amit, and M. Rabani. A large intergenic noncoding RNA induced by p53 mediates global gene repression in the p53 response. *Cell*, 142(3):409–419, 2010. 16, 17

[61] A. Hüttenhofer, M. Kiefmann, S. Meier-Ewert, J. O'Brien, H. Lehrach, J.P. Bachellerie, and J. Brosius. RNomics: an experimental approach that identifies 201 candidates for novel, small, non-messenger RNAs in mouse. *The EMBO Journal*, 20(11):2943–2953, 2001. 16, 17

[62] A. Hüttenhofer, P. Schattner, and N. Polacek. Non-coding RNAs: hope or hype? *TRENDS in Genetics*, 21(5):289–297, 2005. 2

[63] B.D. James, G.J. Olsen, and N.R. Pace. Phylogenetic comparative analysis of RNA secondary structure. *Methods in Enzymology*, 180:227–239, 1989. 23

[64] N. R. Jennings. On agent-based software engineering. *Artificial Intelligence*, 117(2):277–296, 2000. 25

[65] N. R. Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4):35–41, 2001. 29

[66] JRuleEngine, 2014. http://sourceforge.net/. 46, 47, 48

[67] C. L. Kingsford, K. Ayanbule, and S. L. Salzberg. Rapid, accurate, computational discovery of Rho-independent transcription terminators illuminates their relationship to DNA uptake. *Genome Biology*, 8(2):R22, 2007. 14

[68] R. Klein and S. Eddy. RSEARCH: finding homologs of single structured RNA sequences. *BMC Bioinformatics*, 4(1):44, 2003. 19

[69] J. Koolman and K. H. Roehm. *Color Atlas of Biochemistry*. Georg Thieme Verlag, 2005. 4

[70] C. Kutter, H. Schöb, M. Stadler, F. Meins, and A. Si-Ammour. MicroRNA-mediated regulation of stomatal development in Arabidopsis. *The Plant Cell Online*, 19(8):2417–2429, 2007. 13

[71] K. Lagesen, P. Hallin, E.A. Rødland, H.H. Stærfeldt, T. Rognes, and D.W. Ussery. RNAmmer: consistent and rapid annotation of ribosomal RNA genes. *Nucleic Acids Research*, 35(9):3100–3108, 2007. 21

[72] E.S. Lander, L.M. Linton, B. Birren, C. Nusbaum, M.C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001. 1

[73] D. Langenberger, C. Bermudez-Santana, P. F. Stadler, and S. Hoffmann. Identification and classification of small RNAs in transcriptome sequence data. In *Pacific Symposium on Biocomputing*, volume 15, pages 80–87. World Scientific, 2010. 16, 17

[74] D. Laslett and B. Canback. ARAGORN, a program to detect tRNA genes and tmRNA genes in nucleotide sequences. *Nucleic Acids Research*, 32(1):11–16, 2004. 20

[75] A. Lesk. *Introduction to Bioinformatics*. Oxford University Press, 2013. 10

[76] L. Lestrade and M.J. Weber. snoRNA-LBME-db, a comprehensive database of human H/ACA and C/D box snoRNAs. *Nucleic Acids Research*, 34(suppl 1):D158–D162, 2006. 22

[77] C. Liu, B. Bai, G. Skogerbø, L. Cai, W. Deng, Y. Zhang, D. Bu, Y. Zhao, and R. Chen. NONCODE: an integrated knowledge database of non-coding RNAs. *Nucleic Acids Research*, 33(suppl 1):D112–D115, 2005. 1, 21

[78] J. Liu, J. Gough, and B. Rost. Distinguishing Protein-Coding from Non-Coding RNAs through support vector machines. *PLOS Genetics*, 2(4):529–536, 2006. 10, 18

[79] X. Liu, D. Li, W. Zhang, M. Guo, and Q. Zhan. Long non-coding RNA gadd7 interacts with TDP-43 and regulates Cdk6 mRNA decay. *The EMBO Journal*, 31(23):4415–4427, 2012. 17

[80] H. Lodish, A. Berk, and P. Matsudaira. *Molecular Cell Biology*. W. H. Freeman & Co, 2005. 4, 5, 6, 8, 9

[81] S. Lopes. *Introdução à Biologia e origem da vida, Citologia, Reprodução e Embriologia, Histologia*. Saraiva, 1998. 5, 9

[82] Ronny Lorenz, Christoph Flamm, and Ivo L Hofacker. 2d projections of RNA folding landscapes. *GCB*, 157:11–20, 2009. 21

[83] T.M. Lowe and S.R. Eddy. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Research*, 25(5):955–964, 1997. 20, 21

[84] I. Lozada-Chávez, P. F. Stadler, and S. J. Prohaska. Hypothesis for the modern RNA world: a pervasive non-coding RNA-based genetic regulation is a prerequisite for the emergence of multicellular complexity. *Origins of Life and Evolution of Biospheres*, 41(6):587–607, 2011. 13

[85] R. L. Lundblad. *Biochemistry and Molecular Biology Compendium*. Taylor & Francis Group, first edition, 2007. 9

[86] A. Machado-Lima, H.A. del Portillo, and A.M. Durham. Computational methods in noncoding RNA research. *Journal of Mathematical Biology*, 56(1):15–49, 2008. 51

[87] A. Matsuyama, R. Arai, Y. Yashiroda, A. Shirai, A. Kamata, S. Sekido, Y. Kobayashi, A. Hashimoto, M. Hamamoto, and Y. Hiraoka. ORFeome cloning and global analysis of protein localization in the fission yeast Schizosaccharomyces pombe. *Nature Biotechnology*, 24(7):841–847, 2006. 68

[88] J. S Mattick. Non-coding RNAs: the architects of eukaryotic complexity. *EMBO Reports*, 2(11):986–991, 2001. 12

[89] J.S. Mattick and M.J. Gagen. The evolution of controlled multitasked gene networks: the role of introns and other noncoding RNAs in the development of complex organisms. *Molecular Biology and Evolution*, 18(9):1611–1630, 2001. 12

[90] J. Mayfield, Y. Labrou, and T. Finin. Evaluation of KQML as an agent communication language. In *Intelligent Agents II Agent Theories, Architectures, and Languages*, pages 347–360. Springer, 1996. 33

[91] J.S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29(6-7):1105–1119, 1990. 20, 21

[92] J.T. Mendell. Micrornas: critical regulators of development, cellular physiology and malignancy. *Cell Cycle*, 4(9):1179–1184, 2005. 13, 16, 17

[93] T. R. Mercer, M. E. Dinger, and J. S. Mattick. Long non-coding RNAs: insights into functions. *Nature Reviews Genetics*, 10(3):155–159, 2009. 16

[94] T.R. Mercer, M.E. Dinger, S.M. Sunkin, M.F. Mehler, and J.S. Mattick. Specific expression of long noncoding RNAs in the mouse brain. *Proceedings of the National Academy of Sciences*, 105(2):716, 2008. 12

[95] P. Michalak. RNA world–the dark matter of evolutionary genomics. *Journal of Evolutionary Biology*, 19(6):1768–1774, 2006. 1

[96] F. Mignone and G. Pesole. Discrimination of Non-Protein-Coding Transcripts from Protein-Coding mRNA. *RNA Biology Journal*, 2006. 18

[97] A. A. Millar and P. M. Waterhouse. Plant and animal microRNAs: similarities and differences. *Functional & Integrative Genomics*, 5(3):129–135, 2005. 13

[98] miRBase, 2012. miRBase - http://microrna.sanger.ac.uk/. 22

[99] S. A. Mitra, A. P. Mitra, and T. J. Triche. A central role for long non-coding rna in cancer. *Genomic "dark matter": implications for understanding human disease mechanisms, diagnostics, and cures*, page 70, 2012. 17

[100] T. Mituyama, K. Yamada, E. Hattori, H. Okida, Y. Ono, G. Terai, A. Yoshizawa, T. Komori, and K. Asai. The Functional RNA Database 3.0: databases to support mining and annotation of functional RNAs. *Nucleic Acids Research*, 37(suppl 1):D89–D92, 2009. 22

[101] V. A. Moran, R. J. Perera, and A. M. Khalil. Emerging functional and mechanistic paradigms of mammalian long non-coding RNAs. *Nucleic Acids Research*, 40(14):6391–6400, 2012. 17

[102] S. Mukhopadhyay, S. Peng, R. Raje, M. Palakal, and J. Mostafa. Multi-agent information classification using dynamic acquaintance lists. *Journal of the American Society for Information Science and Technology*, 54(10):966–975, 2003. 31

[103] J. C. Nemecek, M. Wüthrich, and B. S. Klein. Global control of dimorphism and virulence in fungi. *Science*, 312(5773):583–588, 2006. 65

[104] M. D. Nodine and D. P. Bartel. Micrornas prevent precocious gene expression and enable pattern formation during plant embryogenesis. *Genes & Development*, 24(23):2678–2692, 2010. 13

[105] NONCODE, 2012. http://www.noncode.org. 21, 22

[106] H. S. Nwana, D. T. Ndumu, L. C. Lee, and J. C. Collis. Zeus: a toolkit for building distributed multiagent systems. *Applied Artificial Intelligence*, 13(1-2):129–185, 1999. 44

[107] S. Ostergaard, L. Olsson, and J. Nielsen. Metabolic engineering of Saccharomyces cerevisiae. *Microbiology and Molecular Biology Reviews*, 64(1):34–50, 2000. 62

[108] K.C. Pang, M.C. Frith, and J.S. Mattick. Rapid evolution of noncoding RNAs: lack of conservation does not mean lack of function. *Trends in Genetics*, 22(1):1–5, 2006. 18

[109] K.C. Pang, S. Stephen, M.E. Dinger, P.G. Engström, B. Lenhard, and J.S. Mattick. RNAdb 2.0? an expanded database of mammalian non-coding RNAs. *Nucleic Acids Research*, 35(suppl 1):D178–D182, 2007. 22

[110] A. Pauli, J. L. Rinn, and A. F. Schier. Non-coding RNAs as regulators of embryogenesis. *Nature Reviews Genetics*, 12(2):136–149, 2011. 12

[111] A. Pavesi, F. Conterio, A. Bolchi, G. Dieci, and S. Ottonello. Identification of new eukaryotic tRNA genes in genomic DNA databases by a multistep weight matrix anaylsis of transcriptional control regions. *Nucleic Acids Research*, 22(7):1247–1256, 1994. 20

[112] M. Pavon-Eternod, S. Gomes, R. Geslain, Q. Dai, M.R. Rosner, and T. Pan. tRNA over-expression in breast cancer and functional consequences. *Nucleic Acids Research*, 37(21):7268–7280, 2009. 14, 15, 16, 17

[113] V. A Petrushin. Emotion recognition agents in real world. In *AAAI fall symposium on socially intelligent agents: human in the loop*, pages 136–138, 2000. 28

[114] A. Pokahr, L. Braubach, and A. Walczak. Jadex user guide. *Hamburg, Germany*, 2007. 43

[115] C. P. Ponting, P. L. Oliver, and W. Reik. Evolution and functions of long noncoding RNAs. *Cell*, 136(4):629–641, 2009. 13, 16

[116] C. Presutti, J. Rosati, S. Vincenti, and S. Nasi. Non-coding RNA and brain. *BMC Neuroscience*, 7(Suppl 1):S5, 2006. 12

[117] C. G. Ralha, M. E. M. T. Walter, M. M. Brigido, and H. W. Schneider. A multi-agent tool to annotate biological sequences. *3rd International Conference on Agents and Artificial Intelligence (ICAART). Rome, Italy. The SciTePress Digital Library: The*

Institute for Systems and Technologies of Information, Control and Communication (INSTICC), pages p. 1–6., 2011. 1, 53

[118] A. Reynolds, D. Leake, Q. Boese, S. Scaringe, W.S. Marshall, and A. Khvorova. Rational siRNA design for RNA interference. *Nature Biotechnology*, 22(3):326–330, 2004. 12

[119] Rfam, 2015. Rfam database - ftp://ftp.sanger.ac.uk/pub/databases/Rfam. 19, 22

[120] E. Rivas and S. Eddy. Noncoding RNA gene detection using comparative sequence analysis. *BMC Bioinformatics*, 2(1):1–8, 2001. 2, 18

[121] RNAdb, 2012. RNAdb - http://jsm-research.imb.uq.edu.au/rnadb. 22

[122] S. J. Russell and P. Norvig. *Artificial intelligence: A Modern Approach. Third edition.* 2010. 24, 25, 26, 27, 28, 31, 32, 45

[123] Y. Sakakibara, M. Brown, R. Hughey, I.S. Mian, K. Sjölander, R.C. Underwood, and D. Haussler. Stochastic context-free grammers for tRNA modeling. *Nucleic Acids Research*, 22(23):5112–5120, 1994. 19

[124] G. San-Blas. Paracoccidioidomycosis and its etiologic agent paracoccidioides brasiliensis. *Medical Mycology*, 31(2):99–113, 1993. 65

[125] G. San-Blas, G. Niño-Vega, and T. Iturriaga. Paracoccidioides brasiliensis and Paracoccidioidomycosis: molecular approaches to morphogenesis, diagnosis, epidemiology, taxonomy and genetics. *Medical Mycology*, 40(3):225–242, 2002. 65

[126] A. Saxena and P. Carninci. Long non-coding RNA modifies chromatin. *Bioessays*, 33(11):830–839, 2011. 17

[127] H. W. Schneider. Implementação de protótipo de um SMA para anotação manual em projetos de sequenciamento de genomas. *Monografia de Graduação do Departamento de Ciência da Computação da Universidade de Brasília*, 2006. 1, 53

[128] H. W. Schneider. Método de aprendizagem por reforço no sistema BioAgents. *Dissertação de Mestrado do Departamento de Ciência da Computação da Universidade de Brasília*, 2010. 1

[129] P. Schuster, P.F. Stadler, and A. Renner. RNA structures and folding: from conventional to new issues in structure predictions. *Current opinion in structural biology*, 7(2):229–235, 1997. 23

[130] J. R. Searle. *Speech Acts: An Essay in the Philosophy of Language*, volume 626. Cambridge University Press, 1969. 33, 34

[131] SeSam, 2012. Sesam - http://www.simsesam.de. 2

[132] J.C. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology.* PWS Pub., 1997. x, 1, 10, 11, 17

[133] N. P. Silva and L. E. C. Andrade. Noções básicas de Biologia Molecular. *Revista Brasileira de Reumatologia*, 41:83–94, 2001. 5, 9, 10

[134] R. Smith. The contract net protocol: Highlevel communication and control in a distributed problem solver. *IEEE Trans. on Computers*, 29:1–12, 1980. 35

[135] Plant snoRNA, 2012. Plant snoRNA database - http://bioinf.scri.sari.ac.uk/cgi-bin/plant_snorna/home. 22

[136] snoRNAbase, 2012. snoRNAbase - http://www-snorna.biotoul.fr/. 22

[137] G. Soldà, I.V. Makunin, O.U. Sezerman, A. Corradin, G. Corti, and A. Guffanti. An Ariadne's thread to the identification and annotation of noncoding RNAs in eukaryotes. *Briefings in Bioinformatics*, 10(5):475–489, 2009. 21

[138] L. Stryer, L. Berg, and J. L. Tymoczco. *Biochemistry*. W. H. Freeman & Co, fifth edition, 2002. 5, 16, 17, 51

[139] M. M. Teixeira, R. C. Theodoro, G. Nino-Vega, E. Bagagli, and M. S.S. Felipe. Paracoccidioides species complex: Ecology, phylogeny, sexual reproduction, and virulence. *PLoS Pathogens*, 10(10):e1004397, 2014. 66

[140] I. Tinoco Jr and C. Bustamante. How RNA folds. *Journal of Molecular Biology*, 293(2):271–281, 1999. 15

[141] E. Torarinsson, M. Sawera, J.H. Havgaard, M. Fredholm, and J. Gorodkin. Thousands of corresponding human and mouse genomic regions unalignable in primary sequence contain common RNA structure. *Genome Research*, 16(7):885–889, 2006. 17, 18

[142] A. Torres-Larios, K.K. Swinger, A.S. Krasilnikov, T. Pan, and A. Mondragón. Crystal structure of the RNA component of bacterial ribonuclease P. *Nature*, 437(7058):584–587, 2005. 23

[143] S. Valadkhan. snRNAs as the catalysts of pre-mRNA splicing. *Current Opinion in Chemical Biology*, 9(6):603–608, 2005. 17

[144] J.C. Venter, M.D. Adams, E.W. Myers, P.W. Li, R.J. Mural, G.G. Sutton, H.O. Smith, M. Yandell, C.A. Evans, R.A. Holt, et al. The sequence of the human genome. *Science's STKE*, 291(5507):1304–1351, 2001. 1

[145] D. Voet and J. G. Voet. *Biochemistry*. Wiley, second edition, 1995. 10

[146] C. Wahlestedt. Natural antisense and noncoding RNA transcripts as potential drug targets. *Drug Discovery Today*, 11(11):503–508, 2006. 18

[147] C. Wang, C. Ding, R.F. Meraz, and S.R. Holbrook. PSoL: a positive sample only learning algorithm for finding non-coding RNA genes. *Bioinformatics*, 22(21):2590–2596, 2006. 18

[148] O. Wapinski and H. Y. Chang. Long noncoding RNAs and human disease. *Trends in Cell Biology*, 21(6):354–361, 2011. 17

[149] S. Washietl, I.L. Hofacker, and P.F. Stadler. Fast and reliable prediction of noncoding RNAs. *Proceedings of the National Academy of Sciences of the United States of America*, 102(7):2454–2459, 2005. 55

[150] J.D. Watson and F.H.C. Crick. Molecular structure of nucleic acids. *Nature*, 171(4356):737–738, 1953. 1

[151] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT press, 1999. 37, 40, 41

[152] D. Weyns and T. Holvoet. A formal model for situated multi-agent systems. *Fundamenta Informaticae*, 63(2):125–158, 2004. 31

[153] N. Williams. Yeast genome sequence ferments new research. *Science*, 272(5261):481–481, 1996. 63

[154] M. Winikoff. Jack^{TM} intelligent agents: An industrial strength platform. In *Multi-Agent Programming*, pages 175–193. Springer, 2005. 42

[155] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning tools and Techniques*. Morgan Kaufmann, 2005. 63

[156] V. Wood, R. Gwilliam, M.A. Rajandream, M. Lyne, R. Lyne, A. Stewart, J. Sgouros, N. Peat, J. Hayles, and S. Baker. The genome sequence of Schizosaccharomyces pombe. *Nature*, 415(6874):871–880, 2002. 67

[157] M. Wooldridge. *An Introduction to MultiAgent Systems*. Second edition, 2009. 2, 24, 25, 28, 29, 37, 39, 41, 42

[158] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(02):115–152, 1995. 39

[159] M. J. Wooldridge and N. R. Jennings. Agent Theories, Architectures and Languages: A Survey. pages 1–39, 1994. 37

[160] C. Xue, F. Li, T. He, G.P. Liu, Y. Li, and X. Zhang. Classification of real and pseudo microRNA precursors using local structure-sequence features and support vector machine. *BMC Bioinformatics*, 6(1):310, 2005. 18

[161] M. Zuker and D. Sankoff. RNA secondary structures and their prediction. *Bulletin of Mathematical Biology*, 46(4):591–621, 1984. 20

[162] M. Zuker and P. Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Research*, 9(1):133–148, 1981. 20, 21, 52

Apêndice A

417 ncRNAs da *Saccharomyces cerevisiae*

A tabela a seguir mostra uma comparação entre a anotação feita pelo sistema ncRNA-Agents e Infernal, para um total de 417 ncRNAs (conjunto positivo) do fungo *Saccharomyces cerevisiae*.

Table 1: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	**Start position**	**Stop position**	**Type**	**Strand**	**NcRNA-Agents annotation**	**Infernal annotation**
I	99305	99868	ncRNA	+	Not annotated	not annotated
I	142367	142468	snoRNA	+	SNORD18	SNORD18
I	166267	166339	tRNA	+	tRNA	tRNA
I	181141	181178	tRNA	+	tRNA	tRNA
I	139152	139187	tRNA	+	tRNA	tRNA
I	182603	182522	tRNA	-	tRNA	tRNA
II	681862	680688	snRNA	-	U2	U2
II	307587	308887	telomerase_RNA	+	Sacc_telomerase	Sacc_telomerase
II	307345	307185	snoRNA	-	snR161	snR161
II	88190	88277	snoRNA	+	snR56	snR56
II	643078	643007	tRNA	-	tRNA	tRNA
II	405960	406031	tRNA	+	tRNA	tRNA
II	645167	645238	tRNA	+	tRNA	tRNA
II	36398	36434	tRNA	+	tRNA	tRNA
II	197699	197629	tRNA	-	tRNA	tRNA
II	197494	197567	tRNA	+	tRNA	tRNA
II	9583	9666	tRNA	+	tRNA	tRNA
II	347603	347686	tRNA	+	tRNA	tRNA
II	350827	350898	tRNA	+	tRNA	tRNA
II	405878	405949	tRNA	+	tRNA	tRNA
II	227075	227156	tRNA	+	tRNA	tRNA
II	266378	266450	tRNA	+	tRNA	tRNA
II	326865	326792	tRNA	-	tRNA	tRNA
III	178798	178610	snoRNA	-	snR189	snR189
III	142546	142364	snoRNA	-	snR33	snR33
III	107712	107504	snoRNA	-	snR43	snR43
III	177183	177282	snoRNA	+	snR65	snR65
III	82533	82462	tRNA	-	tRNA	tRNA
III	142771	142701	tRNA	-	tRNA	tRNA
III	151356	151284	tRNA	-	tRNA	tRNA
III	90859	90896	tRNA	+	tRNA	tRNA
III	149920	149991	tRNA	+	tRNA	tRNA
III	127716	127789	tRNA	+	tRNA	tRNA
III	123648	123577	tRNA	-	tRNA	tRNA
III	168372	168301	tRNA	-	tRNA	tRNA
III	227942	227978	tRNA	+	tRNA	tRNA
III	295484	295556	tRNA	+	tRNA	tRNA
IV	1402919	1403042	snoRNA	+	snR13	snR13
IV	541700	541602	snoRNA	-	snR47	snR47
IV	323471	323217	snoRNA	-	snR63	snR63
IV	1493026	1492477	snoRNA	-	snR84	snR84
IV	410379	410451	tRNA	+	tRNA	tRNA
IV	568964	569035	tRNA	+	tRNA	tRNA
IV	1017278	1017207	tRNA	-	tRNA	tRNA
IV	1095405	1095370	tRNA	-	tRNA	tRNA
IV	1257079	1257008	tRNA	-	tRNA	tRNA
IV	83548	83618	tRNA	+	tRNA	tRNA
IV	992832	992902	tRNA	+	tRNA	tRNA
IV	668080	668007	tRNA	-	tRNA	tRNA

Table 2: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	Start position	Stop position	Type	Strand	NcRNA-Agents annotation	Infernal annotation
IV	884361	884397	tRNA	+	tRNA	tRNA
IV	1201750	1201822	tRNA	+	tRNA	tRNA
IV	1352538	1352466	tRNA	-	tRNA	tRNA
IV	359577	359613	tRNA	+	tRNA	tRNA
IV	1461758	1461715	tRNA	-	tRNA	tRNA
IV	519743	519826	tRNA	+	tRNA	tRNA
IV	1175829	1175901	tRNA	+	tRNA	tRNA
IV	520972	521043	tRNA	+	tRNA	tRNA
IV	645224	645153	tRNA	-	tRNA	tRNA
IV	802802	802731	tRNA	-	tRNA	tRNA
IV	620041	619969	tRNA	-	tRNA	tRNA
IV	568882	568953	tRNA	+	tRNA	tRNA
IV	437772	437853	tRNA	+	tRNA	tRNA
IV	981055	980974	tRNA	-	tRNA	tRNA
IV	1305630	1305712	tRNA	+	tRNA	tRNA
IV	434336	434264	tRNA	-	tRNA	tRNA
IV	1075544	1075472	tRNA	-	tRNA	tRNA
IV	488870	488797	tRNA	-	tRNA	tRNA
IV	1150941	1150842	tRNA	-	tRNA	tRNA
IV	946312	946350	tRNA	+	tRNA	tRNA
V	118035	117667	ncRNA	-	RNaseP_nuc	RNaseP_nuc
V	441987	442508	ncRNA	+	Fungi_SRP	Fungi_SRP
V	322212	322762	ncRNA	+	srg1	srg1
V	167586	167427	snRNA	-	U4	U4
V	424698	424883	snoRNA	+	snR4	snR4
V	431220	431129	snoRNA	-	snR52	snR52
V	61699	61789	snoRNA	+	snoR53	snoR53
V	61352	61433	snoRNA	+	snR67	snR67
V	52320	52150	snoRNA	-	snR80	snR80
V	312095	312023	tRNA	-	tRNA	tRNA
V	177099	177170	tRNA	+	tRNA	tRNA
V	354934	355005	tRNA	+	tRNA	tRNA
V	487402	487331	tRNA	-	tRNA	tRNA
V	61960	61890	tRNA	-	tRNA	tRNA
V	207357	207428	tRNA	+	tRNA	tRNA
V	434612	434541	tRNA	-	tRNA	tRNA
V	443275	443202	tRNA	-	tRNA	tRNA
V	551358	551285	tRNA	-	tRNA	tRNA
V	135497	135425	tRNA	-	tRNA	tRNA
V	435824	435752	tRNA	-	tRNA	tRNA
V	100133	100204	tRNA	+	tRNA	tRNA
V	250286	250357	tRNA	+	tRNA	tRNA
V	131153	131082	tRNA	-	tRNA	tRNA
V	492424	492352	tRNA	-	tRNA	tRNA
V	138666	138737	tRNA	+	tRNA	tRNA
V	86604	86685	tRNA	+	tRNA	tRNA
V	288524	288443	tRNA	-	tRNA	tRNA
V	438700	438773	tRNA	+	tRNA	tRNA
V	469457	469530	tRNA	+	tRNA	tRNA
VI	131503	131061	ncRNA	-	RUF20	RUF20

Table 3: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	Start position	Stop position	Type	Strand	NcRNA-Agents annotation	Infernal annotation
VI	58521	57815	ncRNA	-	RUF21	RUF21
VI	199813	199299	ncRNA	-	Not annotated	not annotated
VI	221714	221967	ncRNA	+	Not annotated	not annotated
VI	204996	204924	tRNA	-	tRNA	tRNA
VI	157951	157916	tRNA	-	tRNA	tRNA
VI	162228	162298	tRNA	+	tRNA	tRNA
VI	181044	180974	tRNA	-	tRNA	tRNA
VI	226760	226688	tRNA	-	tRNA	tRNA
VI	137559	137486	tRNA	-	tRNA	tRNA
VI	101376	101411	tRNA	+	tRNA	tRNA
VI	191557	191513	tRNA	-	tRNA	tRNA
VI	167437	167475	tRNA	+	tRNA	tRNA
VI	210654	210619	tRNA	-	tRNA	tRNA
VII	144120	141898	ncRNA	-	Not annotated	not annotated
VII	35013	33109	ncRNA	-	Not annotated	not annotated
VII	345986	346230	snoRNA	+	snR10	snR10
VII	365251	365163	snoRNA	-	snoZ196	snoZ196
VII	366469	366374	snoRNA	-	snR39	snR39
VII	545370	545566	snoRNA	+	snR46	snR46
VII	609584	609696	snoRNA	+	snosnR48	snosnR48
VII	939672	939459	snRNA	-	U5	U5
VII	939672	939494	snRNA	-	U5	U5
VII	316788	317055	snoRNA	+	snR82	snR82
VII	774349	774421	tRNA	+	tRNA	tRNA
VII	794417	794489	tRNA	+	tRNA	tRNA
VII	707179	707108	tRNA	-	tRNA	tRNA
VII	531610	531681	tRNA	+	tRNA	tRNA
VII	544648	544577	tRNA	-	tRNA	tRNA
VII	328583	328654	tRNA	+	tRNA	tRNA
VII	401598	401527	tRNA	-	tRNA	tRNA
VII	541850	541921	tRNA	+	tRNA	tRNA
VII	440751	440716	tRNA	-	tRNA	tRNA
VII	845649	845719	tRNA	+	tRNA	tRNA
VII	931023	930953	tRNA	-	tRNA	tRNA
VII	779616	779687	tRNA	+	tRNA	tRNA
VII	110696	110625	tRNA	-	tRNA	tRNA
VII	319781	319852	tRNA	+	tRNA	tRNA
VII	739122	739195	tRNA	+	tRNA	tRNA
VII	122269	122341	tRNA	+	tRNA	tRNA
VII	185786	185714	tRNA	-	tRNA	tRNA
VII	876394	876466	tRNA	+	tRNA	tRNA
VII	115488	115524	tRNA	+	tRNA	tRNA
VII	700988	700953	tRNA	-	tRNA	tRNA
VII	205521	205558	tRNA	+	tRNA	tRNA
VII	423092	423129	tRNA	+	tRNA	tRNA
VII	857421	857378	tRNA	-	tRNA	tRNA
VII	700675	700756	tRNA	+	tRNA	tRNA
VII	731137	731210	tRNA	+	tRNA	tRNA
VII	405470	405541	tRNA	+	tRNA	tRNA
VII	828794	828723	tRNA	-	tRNA	tRNA

Table 4: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	Start position	Stop position	Type	Strand	NcRNA-Agents annotation	Infernal annotation
VII	736340	736411	tRNA	+	tRNA	tRNA
VII	561743	561662	tRNA	-	tRNA	tRNA
VII	661749	661820	tRNA	+	tRNA	tRNA
VII	1004216	1004287	tRNA	+	tRNA	tRNA
VII	412367	412294	tRNA	-	tRNA	tRNA
VII	823482	823555	tRNA	+	tRNA	tRNA
VII	73902	73829	tRNA	-	tRNA	tRNA
VII	287385	287350	tRNA	-	tRNA	tRNA
VII	878745	878710	tRNA	-	tRNA	tRNA
VIII	212409	213118	ncRNA	+	Not annotated	not annotated
VIII	214407	215116	ncRNA	+	Not annotated	not annotated
VIII	381540	381727	snoRNA	+	snR32	snR32
VIII	411228	411317	snoRNA	+	snosnR71	snosnR71
VIII	146314	146242	tRNA	-	tRNA	tRNA
VIII	237883	237848	tRNA	-	tRNA	tRNA
VIII	358513	358478	tRNA	-	tRNA	tRNA
VIII	62755	62826	tRNA	+	tRNA	tRNA
VIII	388928	388893	tRNA	-	tRNA	tRNA
VIII	134321	134392	tRNA	+	tRNA	tRNA
VIII	133107	133026	tRNA	-	tRNA	tRNA
VIII	116179	116107	tRNA	-	tRNA	tRNA
VIII	466990	467061	tRNA	+	tRNA	tRNA
VIII	85371	85298	tRNA	-	tRNA	tRNA
VIII	475778	475706	tRNA	-	tRNA	tRNA
IX	397082	393884	ncRNA	-	mRNAlike lncRNA	not annotated
IX	395999	396939	ncRNA	+	mRNAlike lncRNA	not annotated
IX	97111	97246	snoRNA	+	snR68	snR68
IX	324374	324303	tRNA	-	tRNA	tRNA
IX	336420	336349	tRNA	-	tRNA	tRNA
IX	197592	197663	tRNA	+	tRNA	tRNA
IX	370417	370488	tRNA	+	tRNA	tRNA
IX	183513	183440	tRNA	-	tRNA	tRNA
IX	210665	210738	tRNA	+	tRNA	tRNA
IX	300300	300228	tRNA	-	tRNA	tRNA
IX	248850	248931	tRNA	+	tRNA	tRNA
IX	175031	175103	tRNA	+	tRNA	tRNA
IX	325748	325820	tRNA	+	tRNA	tRNA
X	607424	605936	ncRNA	-	Not annotated	not annotated
X	139691	139566	snoRNA	-	SNORD14	SNORD14
X	139950	139761	snoRNA	-	snR190	snR190
X	663749	663942	snoRNA	+	snR3	snR3
X	228479	228094	snoRNA	-	snR37	snR37
X	349233	349130	snoRNA	-	snosnR60_Z15	snosnR60_Z15
X	197385	197313	tRNA	-	tRNA	tRNA
X	204735	204806	tRNA	+	tRNA	tRNA
X	355456	355527	tRNA	+	tRNA	tRNA
X	374495	374424	tRNA	-	tRNA	tRNA
X	541508	541579	tRNA	+	tRNA	tRNA
X	116010	115939	tRNA	-	tRNA	tRNA
X	396796	396726	tRNA	-	tRNA	tRNA

Table 5: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	Start position	Stop position	Type	Strand	NcRNA-Agents annotation	Infernal annotation
X	531828	531898	tRNA	+	tRNA	tRNA
X	414966	415038	tRNA	+	tRNA	tRNA
X	424515	424432	tRNA	-	tRNA	tRNA
X	617919	617956	tRNA	+	tRNA	tRNA
X	391115	391043	tRNA	-	tRNA	tRNA
X	422937	423009	tRNA	+	tRNA	tRNA
X	517884	517813	tRNA	-	tRNA	tRNA
X	233939	234011	tRNA	+	tRNA	tRNA
X	538555	538626	tRNA	+	tRNA	tRNA
X	355374	355445	tRNA	+	tRNA	tRNA
X	374577	374506	tRNA	-	tRNA	tRNA
X	524093	524012	tRNA	-	tRNA	tRNA
X	59172	59100	tRNA	-	tRNA	tRNA
X	378360	378433	tRNA	+	tRNA	tRNA
X	415966	415931	tRNA	-	tRNA	tRNA
X	354244	354282	tRNA	+	tRNA	tRNA
X	542991	542956	tRNA	-	tRNA	tRNA
XI	283185	283279	snoRNA	+	snoR38	snoR38
XI	559366	559016	snoRNA	-	snR42	snR42
XI	38811	38911	snoRNA	+	snosnR64	snosnR64
XI	364776	364876	snoRNA	+	snosnR69	snosnR69
XI	431138	431030	snoRNA	-	snR87	snR87
XI	219895	219967	tRNA	+	tRNA	tRNA
XI	517988	518060	tRNA	+	tRNA	tRNA
XI	513403	513332	tRNA	-	tRNA	tRNA
XI	141089	141018	tRNA	-	tRNA	tRNA
XI	313472	313401	tRNA	-	tRNA	tRNA
XI	202999	203071	tRNA	+	tRNA	tRNA
XI	578965	579001	tRNA	+	tRNA	tRNA
XI	458557	458594	tRNA	+	tRNA	tRNA
XI	84208	84291	tRNA	+	tRNA	tRNA
XI	74624	74697	tRNA	+	tRNA	tRNA
XI	490968	491040	tRNA	+	tRNA	tRNA
XI	162558	162487	tRNA	-	tRNA	tRNA
XI	46806	46735	tRNA	-	tRNA	tRNA
XI	308217	308144	tRNA	-	tRNA	tRNA
XI	379680	379753	tRNA	+	tRNA	tRNA
XI	302918	302953	tRNA	+	tRNA	tRNA
XII	457732	455933	rRNA	-	SSU_rRNA_eukarya	SSU_rRNA_eukarya
XII	466869	465070	rRNA	-	SSU_rRNA_eukarya	SSU_rRNA_eukarya
XII	455181	451786	rRNA	-	PK-G12rRNA	PK-G12rRNA
XII	464318	460923	rRNA	-	PK-G12rRNA	PK-G12rRNA
XII	455181	451786	rRNA	-	SSU_rRNA_eukarya	SSU_rRNA_eukarya
XII	464318	460923	rRNA	-	SSU_rRNA_eukarya	SSU_rRNA_eukarya
XII	459676	459796	rRNA	+	5S_rRNA	5S_rRNA
XII	468813	468931	rRNA	+	5S_rRNA	5S_rRNA
XII	472465	472583	rRNA	+	5S_rRNA	5S_rRNA
XII	482045	482163	rRNA	+	5S_rRNA	5S_rRNA
XII	485697	485815	rRNA	+	5S_rRNA	5S_rRNA
XII	489349	489469	rRNA	+	5S_rRNA	5S_rRNA

Table 6: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	Start position	Stop position	Type	Strand	NcRNA-Agents annotation	Infernal annotation
XII	455571	455414	rRNA	-	5_8S_rRNA	5_8S_rRNA
XII	464708	464551	rRNA	-	5_8S_rRNA	5_8S_rRNA
XII	198784	199389	snoRNA	+	snR30	snR30
XII	899180	899382	snoRNA	+	snR34	snR34
XII	856710	856920	snoRNA	+	snR44	snR44
XII	794794	794697	snoRNA	-	snosnR55	snosnR55
XII	795024	794937	snoRNA	-	snosnR57	snosnR57
XII	366235	366346	snRNA	+	U6	U6
XII	794575	794486	snoRNA	-	snosnR61	snosnR61
XII	348510	348427	snoRNA	-	snR79	snR79
XII	657006	656934	tRNA	-	tRNA	tRNA
XII	214955	214883	tRNA	-	tRNA	tRNA
XII	427132	427203	tRNA	+	tRNA	tRNA
XII	793918	793989	tRNA	+	tRNA	tRNA
XII	797178	797249	tRNA	+	tRNA	tRNA
XII	734802	734875	tRNA	+	tRNA	tRNA
XII	1052144	1052071	tRNA	-	tRNA	tRNA
XII	605335	605300	tRNA	-	tRNA	tRNA
XII	875376	875412	tRNA	+	tRNA	tRNA
XII	628383	628420	tRNA	+	tRNA	tRNA
XII	963055	962972	tRNA	-	tRNA	tRNA
XII	592562	592519	tRNA	-	tRNA	tRNA
XII	732090	732127	tRNA	+	tRNA	tRNA
XII	976056	975983	tRNA	-	tRNA	tRNA
XII	92548	92583	tRNA	+	tRNA	tRNA
XII	448721	448650	tRNA	-	tRNA	tRNA
XII	374355	374427	tRNA	+	tRNA	tRNA
XII	818609	818680	tRNA	+	tRNA	tRNA
XII	168025	167944	tRNA	-	tRNA	tRNA
XII	687932	687859	tRNA	-	tRNA	tRNA
XII	784354	784389	tRNA	+	tRNA	tRNA
XIII	667456	667288	ncRNA	-	Not annotated	not annotated
XIII	91970	92027	ncRNA	+	Not annotated	not annotated
XIII	652275	652532	snoRNA	+	snR11	snR11
XIII	500072	499984	snoRNA	-	SNORD24	SNORD24
XIII	163620	163535	snoRNA	-	snosnR54	snosnR54
XIII	298554	298651	snoRNA	+	snoZ122	snoZ122
XIII	298307	298412	snoRNA	+	snR73	snR73
XIII	298138	298225	snoRNA	+	SNORD27	SNORD27
XIII	297918	298006	snoRNA	+	snR75	snR75
XIII	297725	297833	snoRNA	+	snR76	snR76
XIII	297506	297593	snoRNA	+	snR77	snR77
XIII	297278	297364	snoRNA	+	snR78	snR78
XIII	626349	626654	snoRNA	+	snR83	snR83
XIII	67938	67768	snoRNA	-	snR85	snR85
XIII	763113	762110	snoRNA	-	snR86	snR86
XIII	321219	321147	tRNA	-	tRNA	tRNA
XIII	768441	768369	tRNA	-	tRNA	tRNA
XIII	463554	463625	tRNA	+	tRNA	tRNA
XIII	290801	290872	tRNA	+	tRNA	tRNA

Table 7: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	Start position	Stop position	Type	Strand	NcRNA-Agents annotation	Infernal annotation
XIII	352280	352316	tRNA	+	tRNA	tRNA
XIII	183968	183898	tRNA	-	tRNA	tRNA
XIII	363064	363135	tRNA	+	tRNA	tRNA
XIII	480693	480621	tRNA	-	tRNA	tRNA
XIII	504895	504932	tRNA	+	tRNA	tRNA
XIII	572955	572883	tRNA	-	tRNA	tRNA
XIII	196103	196068	tRNA	-	tRNA	tRNA
XIII	808317	808246	tRNA	-	tRNA	tRNA
XIII	747963	747892	tRNA	-	tRNA	tRNA
XIII	131896	131825	tRNA	-	tRNA	tRNA
XIII	259239	259158	tRNA	-	tRNA	tRNA
XIII	372518	372445	tRNA	-	tRNA	tRNA
XIII	420661	420588	tRNA	-	tRNA	tRNA
XIII	586709	586636	tRNA	-	tRNA	tRNA
XIII	379338	379303	tRNA	-	tRNA	tRNA
XIII	168795	168833	tRNA	+	tRNA	tRNA
XIII	837928	837966	tRNA	+	tRNA	tRNA
XIV	585587	585926	snoRNA	+	RNase_MRP	RNase_MRP
XIV	230672	230105	snRNA	-	U1_yeast	U1_yeast
XIV	722211	721938	snoRNA	-	snR191	snR191
XIV	89210	89306	snoRNA	+	snR40	snR40
XIV	716120	716284	snoRNA	+	snR49	snR49
XIV	586090	586175	snoRNA	+	snosnR66	snosnR66
XIV	519169	519099	tRNA	-	Not annotated	not annotated
XIV	374869	374905	tRNA	+	tRNA	tRNA
XIV	96312	96241	tRNA	-	tRNA	tRNA
XIV	569940	569867	tRNA	-	tRNA	tRNA
XIV	602312	602385	tRNA	+	tRNA	tRNA
XIV	443006	443043	tRNA	+	tRNA	tRNA
XIV	726217	726134	tRNA	-	tRNA	tRNA
XIV	102716	102789	tRNA	+	tRNA	tRNA
XIV	632599	632672	tRNA	+	tRNA	tRNA
XIV	631917	631846	tRNA	-	tRNA	tRNA
XIV	547094	547129	tRNA	+	tRNA	tRNA
XIV	568115	568150	tRNA	+	tRNA	tRNA
XIV	104805	104877	tRNA	+	tRNA	tRNA
XIV	560765	560693	tRNA	-	tRNA	tRNA
XV	780107	780120	snoRNA	+	Fungi_U3	Fungi_U3
XV	842182	841958	snoRNA	-	snR31	snR31
XV	759529	759326	snoRNA	-	snR35	snR35
XV	680866	680685	snoRNA	-	snR36	snR36
XV	842403	842606	snoRNA	+	snR5	snR5
XV	259489	259578	snoRNA	+	snR50	snR50
XV	136183	136088	snoRNA	-	snR58	snR58
XV	409864	409765	snoRNA	-	snR62	snR62
XV	832332	832521	snoRNA	+	snR8	snR8
XV	234346	234546	snoRNA	+	snR81	snR81
XV	408134	407948	snoRNA	-	snR9	snR9
XV	854259	854187	tRNA	-	tRNA	tRNA
XV	571958	572029	tRNA	+	tRNA	tRNA

Table 8: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	Start position	Stop position	Type	Strand	NcRNA-Agents annotation	Infernal annotation
XV	594425	594354	tRNA	-	tRNA	tRNA
XV	226681	226611	tRNA	-	tRNA	tRNA
XV	282164	282234	tRNA	+	tRNA	tRNA
XV	111033	110962	tRNA	-	tRNA	tRNA
XV	438643	438679	tRNA	+	tRNA	tRNA
XV	710201	710272	tRNA	+	tRNA	tRNA
XV	976421	976493	tRNA	+	tRNA	tRNA
XV	228331	228404	tRNA	+	tRNA	tRNA
XV	487439	487512	tRNA	+	tRNA	tRNA
XV	301097	301132	tRNA	+	tRNA	tRNA
XV	464450	464485	tRNA	+	tRNA	tRNA
XV	980683	980718	tRNA	+	tRNA	tRNA
XV	340371	340299	tRNA	-	tRNA	tRNA
XV	274673	274709	tRNA	+	tRNA	tRNA
XV	113802	113874	tRNA	+	tRNA	tRNA
XV	354113	354041	tRNA	-	tRNA	tRNA
XV	663885	663812	tRNA	-	tRNA	tRNA
XV	288192	288230	tRNA	+	tRNA	tRNA
XVI	281373	281056	snoRNA	-	Fungi_U3	Fungi_U3
XVI	719242	719148	snoRNA	-	snR41	snR41
XVI	821732	821903	snoRNA	+	snR45	snR45
XVI	718806	718700	snoRNA	-	snR51	snR51
XVI	173827	173904	snoRNA	+	snoZ196	snoZ196
XVI	719050	718887	snoRNA	-	snR70	snR70
XVI	856902	856974	tRNA	+	tRNA	tRNA
XVI	435964	435893	tRNA	-	tRNA	tRNA
XVI	775836	775765	tRNA	-	tRNA	tRNA
XVI	210263	210192	tRNA	-	tRNA	tRNA
XVI	560233	560198	tRNA	-	tRNA	tRNA
XVI	622575	622540	tRNA	-	tRNA	tRNA
XVI	572269	572339	tRNA	+	tRNA	tRNA
XVI	860379	860449	tRNA	+	tRNA	tRNA
XVI	819529	819602	tRNA	+	tRNA	tRNA
XVI	880369	880296	tRNA	-	tRNA	tRNA
XVI	582062	582134	tRNA	+	tRNA	tRNA
XVI	769242	769207	tRNA	-	tRNA	tRNA
XVI	338919	338848	tRNA	-	tRNA	tRNA
XVI	810676	810749	tRNA	+	tRNA	tRNA
XVI	689565	689646	tRNA	+	tRNA	tRNA
XVI	744284	744355	tRNA	+	tRNA	tRNA
XVI	56204	56169	tRNA	-	tRNA	tRNA
Mito	6546	8194	rRNA	+	SSU_rRNA_bacteria	SSU_rRNA_bacteria
Mito	58009	60724	rRNA	+	Not annotated	not annotated
Mito	85295	85777	ncRNA	+	Not annotated	not annotated
Mito	69846	69921	tRNA	+	tRNA	tRNA
Mito	64415	64490	tRNA	+	tRNA	tRNA
Mito	68322	68396	tRNA	+	tRNA	tRNA
Mito	35373	35444	tRNA	+	tRNA	tRNA
Mito	77431	77505	tRNA	+	tRNA	tRNA
Mito	67468	67542	tRNA	+	tRNA	tRNA

Table 9: Comparação entre anotao ncRNA-Agents e Infernal - 417 ncRNAs

Chr	**Start position**	**Stop position**	**Type**	**Strand**	**NcRNA-Agents annotation**	**Infernal annotation**
Mito	64596	64670	tRNA	+	tRNA	tRNA
Mito	70162	70237	tRNA	+	tRNA	tRNA
Mito	67061	67134	tRNA	+	tRNA	tRNA
Mito	66095	66179	tRNA	+	tRNA	tRNA
Mito	72630	72705	tRNA	+	tRNA	tRNA
Mito	85035	85112	tRNA	+	tRNA	tRNA
Mito	71433	71503	tRNA	+	tRNA	tRNA
Mito	731	802	tRNA	+	tRNA	tRNA
Mito	66210	66285	tRNA	+	tRNA	tRNA
Mito	69289	69362	tRNA	+	tRNA	tRNA
Mito	67309	67381	tRNA	+	tRNA	tRNA
Mito	69203	69288	tRNA	+	Not annotated	not annotated
Mito	48201	48290	tRNA	+	tRNA	tRNA
Mito	78162	78089	tRNA	-	tRNA	tRNA
Mito	63862	63937	tRNA	+	tRNA	tRNA
Mito	78533	78608	tRNA	+	tRNA	tRNA
Mito	9374	9447	tRNA	+	tRNA	tRNA
Mito	70824	70907	tRNA	+	Not annotated	not annotated

Apêndice B

Artigo BSB 2013

Artigo aceito no *8th Brazilian Symposium on Bioinformatics* - BSB, estrato B4 em Conferência no Qualis.

ncRNA-Agents: a multiagent system for non-coding RNA annotation

Wosley Arruda[1], Célia G. Ralha[1], Tainá Raiol[2], Marcelo M. Brígido[2], Maria Emilia M. T. Walter[1], Peter F. Stadler[3]

[1] Department of Computer Science, University of Brasília, Brazil
[2] Laboratory of Molecular Biology, Institute of Biology, University of Brasília, Brazil
[3] Department of Computer Science and Interdisciplinary Center for Bioinformatics, University of Leipzig, Germany

Abstract In recent years, non-coding RNAs (ncRNAs) have been focus of intensive research. Since the characteristics and signals of ncRNAs are not entirely known, researchers use different computational tools together with their biological knowledge to predict potential ncRNAs. In this context, this work presents a multiagent system to annotate ncRNAs based on the output of different tools, using inference rules to simulate biologists' reasoning. Experiments with real data of fungi allowed to identify novel putative ncRNAs, which shows the usefulness of our approach.

1 Introduction

Since 1990, important roles of the RNA molecules have been identified, besides the well defined functions of messenger RNA (mRNA), transporter RNA (tRNA) and ribosomal RNA (rRNA), all involved in protein synthesis. Non-coding RNAs (ncRNAs), those RNAs not coding for proteins [6,25], present specific spatial conformation that allow them to play regulatory roles in an extensive variety of biological reactions and processes, e.g., translation initiation, level control of mRNA, stem-cell maintenance, brain developing, metabolism regulation, support to protein transport, nucleotide edition, imprinting regulation and chromatin dynamics [22].

On one side, although intensive efforts worldwide, biological and computational methods are not yet capable to easily identify and classify ncRNAs, directly affecting the annotation of these transcripts. From a biological point of view, ncRNAs are characterized by its transcription and absence of translation into proteins, and RNAs presenting very different nucleotide sequences (primary sequences) but similar spatial conformations (secondary structure) perform the same cellular functions. Therefore, ncRNAs should be characterized by their secondary structures and not only by their primary sequences. From a computational point of view, ncRNAs can not be identified and classified by homology tools, extensively and efficiently used to annotate protein coding genes (e.g., BLAST), and methods designed to predict secondary from primary structure are commonly used to annotate ncRNAs [13,6]. In this context, biologists use

different tools to annotate sequences that appear to be ncRNAs, together with their knowledge, which is a reasoning intensive process.

On the other side, a multiagent system (MAS) is characterized by the distribution of intelligence among autonomous entities called agents, which interact to reach individual or collective goals. In order to do that, the agents in a MAS have to negotiate in a cooperative way, coordinating and joining efforts to reach the objectives, normally not accomplished by a single agent [31].

In this context, this work has the objective of presenting ncRNA-Agents, a MAS to support ncRNA annotation, using inference rules to simulate the biologists' reasoning to combine the output of different tools and data bases [27]. As far as we know, there are no computational tools simulating the biologists' reasoning to annotate ncRNAs.

This work is divided into six sections. In Section 2, challenges to annotate ncRNAs are firstly discussed, together with commonly used tools and data bases. After, a general classification for ncRNA annotation tools is proposed. In Section 3, concepts and tools to implement MAS are introduced. In Section 4, the ncRNA-Agents architecture and some implementation details are detailed. In order to validate ncRNA-Agents, two experiments with real data of fungi are analized in Section 5. Finally, this work is concluded in Section 6.

2 Challenges to annotate ncRNAs

Methods to identify and classify ncRNAs use different strategies, based on ncRNA characteristics supported by *in silico* and experimental findings. Some criteria are commonly used: a number of known ncRNA classes do not present long ORFs; their sequences have unexpected stop codons; RNAs usually present conservation in their secondary (spatial) structure and rarely in their primary (DNA sequences) structure; some known ncRNAs have complex tridimensional structures, and catalyst or structural functions. These characteristics prevent to use traditional similarity based tools to predict proteins [25]. Methods using biological hints are also extensively used to predict ncRNAs: codons, synonymous and non-synonymous substitutions, and minimum folding energy [21].

One important problem when predicting ncRNAs is the lack of available experimental information, confirming *in silico* prediction. Although computational predictions are useful, confirmation needs biological experiments, such as RNAseq of small RNAs, real-time PCR and gene deletion or knock out experiments. But the absence of translation is not conclusive, since predicted ncRNA may not be transcribed or translated when some environmental or physiological conditions happen. Therefore, to predict ncRNAs, biologists use different bioinformatics tools, with distinct methods, and after make a combined analysis of all these information to decide if a sequence is a potential ncRNA. We have three main problems when annotating ncRNAs: secondary structure prediction [14], secondary structure comparison [13,6] and ncRNA identification and classification [10].

2.1 Tools and data base to annotate ncRNAs

In this section, we describe first some computational tools and after data bases [30] commonly used to identify and classify ncRNAs.

BLAST (*Basic Local Alignment Search Tool*) [1] is extensively used to predict proteins. Although Blast can successfully detect some specific ncRNA families (e.g., snoRNAs), in general, only nucleotide comparisons is not enough, since many ncRNA classes are very different regarding to the primary sequences but exhibit a common secondary structure. **Infernal** (INFERence of RNA Alignment) [5] identifies ncRNAs looking for secondary structures. It builds profiles of RNA consensus spatial structure, known as covariance models (CMs), in order to find similarities between the investigated sequence and the consensus secondary structure of each one of the RNA families stored in the Rfam data base [11]. **tRNAscan** [19] is considered one of the more precise tRNA predictors. It combines three programs: two tRNA predictors and a covariance model [5] previously trained with tRNA sequences. The execution of these three programs results is a tRNA identifier presenting high sensibility ($99 - 100\%$) and specificity (with a false positive rate less than 0.00007 by Mb) in a reasonable velocity. **Portrait** [2] identifies ncRNAs of not complete transcriptomes of yet not entirely characterized species, based on Support Vector Machine (SVM). The result of Portrait is a probability indicating the likelihood of a transcript to be non protein coding. **Vienna** [13] is a set of packages used to generate or to compare RNA secondary structures. Folding in this tool uses prediction algorithms based on RNA free energy and the probability of base pairing [21]. Particularly, RNAfold [13,14] package is based on the hypothesis that an RNA molecule is folded in a more stable thermodynamics structure, the one presenting the minimum free energy. **RNAmmer** [16] predicts rRNAs using the 5S ribosomal and the European ribosomal RNA data base to generate many structural alignments, which are used to build Markov chain libraries.

Now, we briefly describe commonly used ncRNA data bases, created from experimental and computational data [30]. **NONCODE** [18] includes many different types of ncRNAs, except tRNAs and rRNAs. More than 80% of the input in NONCODE are experimentally confirmed. We used NONCODE version v3.0, with more than $411,552$ ncRNAs. **RNAdb** [24] contains sequences and annotation of thousands of mammalian ncRNAs, but a large number of them do not have known biological functions. **miRBase** [12] is a data base of microRNAs. **snoRNA** database [17] contains human snoRNAs of both types, H/ACA and C/D box. **fRNAdb** [23] integrates a set of data bases, including NONCODE and RNAdb. **Rfam** [11] is a curated data base containing information about ncRNA families, presenting two classes of data: CM profile and seed alignments for each ncRNA family. In this work, we used Rfam 11.0, with $2,208$ families.

2.2 A proposal to classify ncRNA annotation tools

We propose a general classification for computational tools that annotate ncRNAs, in three groups, described as follows: (i) **Homology**: in this group, ncR-

NAs are predicted using alignment among sequences. These comparisons depend on the quality of the annotation stored in data bases, since this quality affects ncRNA classification. BLAST [1] and Infernal [6] are two extensively used tools, both using the concept of similarity (the greater the similarity the greater the chance that both sequences have conserved the same function). In general, BLAST does not produce good results to annotate ncRNAs, being Infernal more sensitive and specific to identify hundreds of different types of ncRNAs; (ii) **Class prediction**: in this group, ncRNAs are identified using methods based on machine learning. In the supervised learning paradigm, we take a set of known ncRNAs and a set of known proteins, computing *ab initio* characteristics for these sequences, in order to create a model to predict ncRNAs. Examples are PORTRAIT [2] and DARIO [7]; and (iii) ***De novo* model**: in this group, ncRNAs are predicted using models distinct from *homology* and *class prediction*, e.g., the Vienna [13] thermodynamic model.

3 MAS and inference rules

In this section, we briefly describe MAS, as well as inference rules, respectively used for knowledge representation and reasoning. First, we present the concept of an agent. According to Russell and Norvig [29], an agent is an entity that interacts with the environment, perceiving it through its sensors and acting using actuators. An agent has two important characteristics: it is capable to act autonomously, taking decisions leading to the satisfaction of its objectives; and it is capable to interact with other agents through human inspired social interaction protocols. We may cite as agents functionalities: coordination, cooperation, competition and negotiation.

A MAS includes many homogeneous or heterogeneous agents interacting and working together. Each agent operates asynchronously related to other agents [31]. In order to have agent communication and interaction we need to have an adequate MAS infrastructure with specific agent language and interaction protocol.

JADE (Java Agent DEvelopment Framework) [3] is a commonly used framework to develop MAS, since it follows the patterns of FIPA (Foundation for Intelligent Physical Agents), an international organization responsible to define patterns for the development of agent technologies. JADE allows and facilitates the development of agents using Java language since it has many already defined functions, uses ACL (Agent Communication Language), and has ready to use interaction protocols (e.g., contract net). JADE presents a visual interface to monitor agents' execution that helps to control these agents' life cycle, having also many built-in resources, e.g., the directory facilitator (yellow pages) and the agent controller manager (white pages). In this work, we have used JADE to implement the ncRNA-Agents prototype.

In intelligent agents, reasoning skills can be implemented using inference rules, which also allows to represent knowledge that will be manipulated by the inference engine. Drools [4] is an open source rule engine that allows to

build a knowledge base and make inferences based in patterns. Drools interacts with Java and the knowledge is obtained from a set of declarative rules. A Drools' rule has one or more condition (or facts) leading to one or more actions (or consequences). Basically, the Drools inference engine offers the possibility of using a forward chaining or a backward chaining method as investigation approaches. The inference algorithm of Drools is RETE [9], which is adapted to object oriented systems. In order to integrate the inference rule engine to JADE framework we have used Drools.

4 ncRNA-Agents architecture and prototype

In this section, we first present the ncRNA-Agents architecture, and after details of the implemented prototype. This ncRNA annotation system using MAS was inspired in previous work of our group [26,27].

4.1 Architecture proposal

Figure 1 presents the ncRNA-Agents architecture with four different layers: (i) **interface layer**: receives from a user one sequence (or a sequence file) in FASTA format, together with the required annotation tools. The system will return the RNA annotation (a potential ncRNA or not) of each sequence, along with the used tools and corresponding reasoning; (ii) **conflict resolution layer**: receives the user request and decides the better recommendations based on suggestions received from the collaborative layer, sending them to the interface layer; (iii) **collaborative layer**: execute tools (the user can choose the tools among the ones presented in the ncRNA-Agents initial page) to annotate ncRNAs, sending the obtained results to the conflict resolution layer; and (iv) **physical layer**: stores different data bases, as those presented in Section 2.1.

In Figure 1, note that the collaborative layer present different levels of agents: manager and analyst agents. **Manager agents** are responsible for filtering the suggestions sent by the analyst agents. We defined four types of manager agents, three of them following the classification of the computational methods to annotate ncRNAs proposed in Section 2.2: (i) **homology manager agent**: coordinates agents working on tools based on homology, e.g., BLAST [1] and Infernal [6]; (ii) **class prediction manager**: coordinates agents working on tools based on machine learning, e.g., Portrait [2]; (iii) ***de novo* manager**: coordinates agents working on tools that do not use reference organisms, e.g., RNAfold [14] from Vienna package; and (iv) **seeker manager**: coordinates agents created to refine analyses of the other manager agents (mainly removing false positives), simulating the reasoning of a biologist when analyzing the results of the required programs and data bases. It also gives flexibility to the architecture, since it can be adapted according to the purposes of each project. Finally, **analyst agents** are responsible for executing specific tools to annotate ncRNAs. Each agent (created from a manager agent request) parses the output file generated by the tool controlled by this agent. This analysis is returned as

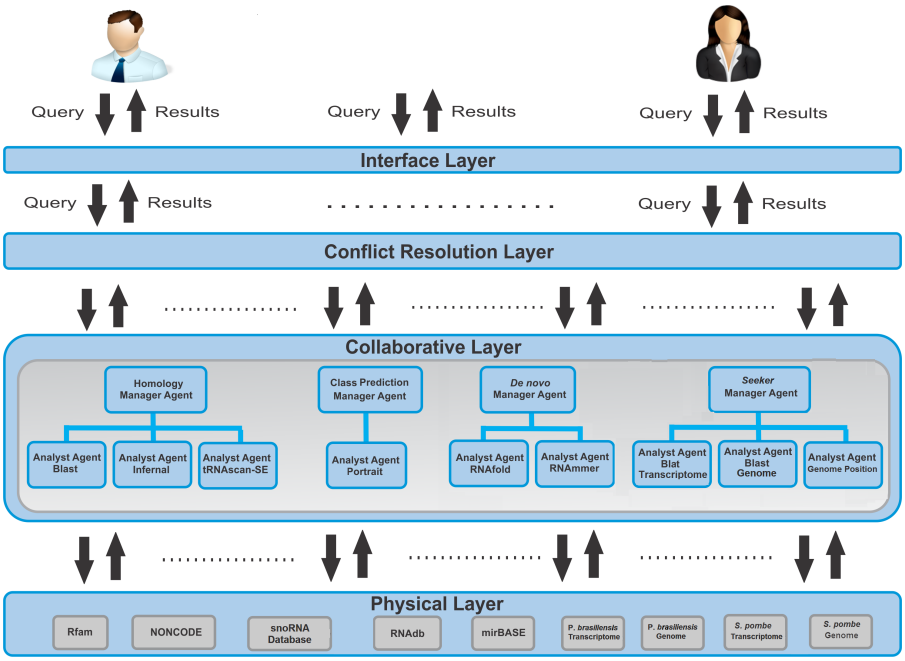


Figure 1. ncRNA-Agents architecture.

an annotation to its corresponding manager agent. Particularly, analyst agents can be created in seeker manager to improve annotation, e.g., to check if a ncRNA candidate belongs to intronic or intergenic region, or to build a multiple alignment for investigating conserved domains among related organisms. This manager allows to remove false positives, e.g, seeker manager can verify if a predicted snoRNA does not belong to an exonic region.

4.2 Prototype implementation

A prototype was created with four manager agents: homology, class prediction, *de novo* and seeker. The ncRNA-Agents system was written to be time efficient using threads executed in parallel. It was implemented with JADE version 4.1 [3]. *Eclipse SDK* version Helios 2012 was used as the development environment together with the web server apache tomcat 7.0. *JADE* was adopted for a couple of reasons. It is a free software distributed under *LGPL* license, and *Java* language allows portability. Since it is a ready to use plataform, it is not necessary to implement agent funcionalities, agent management ontologies and message transport mechanism.

ncRNA-Agents uses *Drools* version 5.5.0 for the agents reasoning [4]. We defined the biological knowledge using declarative rules according to the parameters defined for the experiments. Rules for homology manager were created for

choosing, among the results of the three analysts (Blast, Infernal and tRNAscan), the better annotation, which is sent to the conflict resolution layer. tRNAscan analyst verifies if there are any good alignments, selecting the alignment with the highest score. Blast analyst verifies if there are alignments with e-value $\leq 10^{-5}$ (adopted by the biologists, but this value is an easily changeable system parameter), selecting the lower e-value in this case. Infernal analyst verifies if there are alignments with score ≥ 34, selecting the alignment with the highest score. We note that the analyst agents execute in parallel.

We called *potential ncRNA* an expressed sequence located in an intronic region (considering as intronic those sequences inside a gene presenting an intersection of at most 50% with an exon) or in an intergenic region. To identify such a ncRNA, the rules for seeker manager were designed using three analysts. The first one, Blat analyst, executes Blat with *P. brasiliensis* transcripts [8], in order to verify if there are alignments with e-value $\leq 10^{-5}$, selecting the lowest e-value in this case. This was designed to verify if a gene is expressed (according to Blat analyst). In this case, seeker manager asked the second analyst, Blast PB Genome, to check the location of this sequence in the *P. brasiliensis* genome (downloaded from Broad Institute), while the third analyst, Genome Position, was in charge to verify if the sequence occurs in an intronic or an intergenic region, as described above. Genome Position analyst takes as input a file in .gtf format (also downloaded from Broad Institute) containing the positions of start and stop codons, as well as CDS and exonic regions. We considered a sequence to be located in an intergenic region if it was not located in one of the regions reported in this file but was close enough to a gene ($\leq$ 1.000 bp).

Finally, the conflict resolution layer (CR layer) rules were designed as follows. The first decision is about annotating the sequence as tRNA, if the homology manager sends an annotation identified by tRNAscan analyst. If it is not the case, CR layer decides if the sequence can be annotated as rRNA, if *de novo* Manager sends an annotation identified by RNAmmer analyst. Both tools are considered reliable to annotate tRNAs and rRNAs, respectively. If the sequence could not be annotated as tRNA or rRNA, CR layer verifies if homology manager sent a recommendation based on Infernal analyst or Blast analyst (with databases snoRNA, RNAdb, NONCODE and mirBASE, in this order), sending it to the interface layer if it is the case.

If a recommendation could not be found by homology manager, CR layer verifies if class prediction manager sends a recommendation. If the probability of being a ncRNA is $< 70\%$, the recommendation of the Portrait analyst is "not annotated by Portrait". On the other hand (probability $\geq 70\%$), CR layer validates this with other analyses, as follows. The first verification is done by *de novo* manager. It executes RNAfold analyst, which indicates as potential ncRNA those sequences with *Minimum Free Energy (MFE)* $\leq -5.00\ kcal/mol$. If it was not the case, the recommendation is "not annotated by RNAfold". Otherwise, CR layer calls seeker manager to check the three aspects described above: (i) verifies if the RNA is expressed in the *P. brasiliensis* trancriptome (Blat Analyst); (ii) finds where this RNA is located in the chromosomes of the *P. brasiliensis*

genome (Blast Analyst); and (iii) verifies if this sequence is located in intron, exon or intergenic region.

5 Results

To validate the prototype, we developed two experiments with real data of two fungi: *Paracoccidioides brasiliensis* [8] and *Schizosacharomyces pombe* [20].

Detecting ncRNAs in *Paracoccidioides brasiliensis*

First, 6,022 sequences from *P. brasiliensis* transcriptome [8] were submitted to Blast with SwissProt data base, to annotate proteins. Sequences not identified as proteins were submitted as input to ncRNA-Agents, noting that sequences annotated as "hypothetical protein", "predicted protein", "potential protein" and "unknown sequence" were also included in this input file. For the experiment, BLAST was executed with data bases snoRNA, RNAdb, NONCODE and mirBASE. Infernal was executed with Rfam 11.0 data base, and the *de novo* tools tRNAscan and Portrait were both executed. We also used Blat with *P. brasiliensis* transcripts, Blast with *P. brasiliensis* genome, and a script to verify if a sequence was located in intergenic or intronic regions.

Table 1. ncRNAs in *P. brasiliensis* identified by ncRNA-Agents. Column "Genome Position" indicates if a sequence is in intronic or intergenic region (in this case, the closest gene is indicated).

Sequence Name	SuperContig	Initial Position	Final Position	ncRNA Length (bp)	Strand	Genome Position
Pb_ncRNA_1	Supercontig_1.13	895440	895118	322	-	Intergenic: After PAAG_05248
Pb_ncRNA_2	Supercontig_1.48	5192	5578	386	+	Intron
Pb_ncRNA_3	Supercontig_1.3	248422	248850	428	+	Intergenic: Before PAAG_01391
Pb_ncRNA_4	Supercontig_1.109	3734	3518	216	-	Intron
Pb_ncRNA_5	Supercontig_1.58	25692	26297	605	+	Intergenic: After PAAG_08977
Pb_ncRNA_6	Supercontig_1.39	84839	84453	386	-	Intron
Pb_ncRNA_7	Supercontig_1.33	1559	1734	175	+	Intergenic: Before PAAG_08374
Pb_ncRNA_8	Supercontig_2.66	478	100	378	-	Intergenic: Before PAAG_12001
Pb_ncRNA_9	Supercontig_1.9	14977	15094	117	+	Intergenic: Before PAAG_03665

We found 9 potential ncRNAs, as shown in Table 1. Pb_ncRNA_1, located into a intergenic region, was identified as the Fungi_SRP (Fungal signal recognition particle RNA), the RNA component of the signal recognition particle (SRP) ribonucleoprotein complex [28]. SRP RNA, also known as 7SL, 6S, ffs, or 4.5S RNA, participates in the coordination of protein traffic to cellular membranes. The RNA and protein components of the ribonucleoprotein complex are

highly conserved in all domains of life, but varying the molecule size and the number of RNA secondary structure elements. The eukaryotic SRP consists of a 300-nucleotide 7S RNA and six proteins: SRPs 72, 68, 54, 19, 14 and 9. Although we could not classify Pb_ncRNA_3, it presented a high similarity with the small ncRNA AFU_254 (sRNA Afu-254) found in *Aspergillus fumigatus*, a pathogenic filamentous fungus responsible for infections worldwide, suggesting a conservation of this potential small ncRNA between these two pathogenic fungi.

Detecting ncRNAs in *Schizosacharomyces pombe*

Small RNA-Seq data of *S. pombe* was extracted from EMBL-EBI (experiment E-MTAB-1154). After filtering to remove low quality reads, mapping was performed using Segehmel [15], taking as reference the *S. pombe* genome (downloaded from BROAD Institute). Next, a Perl script identified genome continuous regions presenting at least five mapped reads, and another Perl script found the location of each region in the *.bed* file generated by mapping. The annotation followed the same reasoning designed for *P. brasiliensis*, except for the seeker manager, which only verified if the predicted ncRNA was located in exon, intron or intergenic region (Blat and Blast analysts were not used here). The first script was executed in the three chromosomes of *S. pombe*, having identified respectively $9,566$, $7,565$ and $4,126$ regions. These regions were submitted to ncRNA-Agents, which identified 52 putative ncRNAs, all in strand "+". General features of some of those ncRNAs are shown in Table 2.

Sequences not annotated as ncRNAs by homology, class prediction and *de novo* managers were submitted only to seeker manager. This allows to find other 2,493 putative ncRNAs (728 in exons and 1,765 in introns) using the annotation of the reference .gtf file: (i) in chromosome I, 327 in exons (7 tRNAs, 2 snoRNAs, 1 snRNA, 317 putative ncRNAs) and 834 in introns; (ii) in chromosome II, 276 in exons (17 tRNAs, 9 rRNAs, 4 snRNAs, 246 putative ncRNAs) and 618 in introns; and (iii) in chromosome III, 125 in exons (9 tRNAs, 4 rRNAs, 1 snoRNA, 111 putative ncRNAs) and 313 in introns.

It is important to note that, in both cases (predicted ncRNAs obtained from all the managers working together and those by seeker manager only), to be predicted as putative ncRNAs, a sequence had to present folding free energy < -5.00 kcal/mol (according to RNAfold), indicating likely stable conformational structures. Figure 2 shows some examples.

6 Conclusion

In this article, we presented ncRNA-Agents, a multiagent system to support ncRNA annotation. The proposed architecture allows the execution of intelligent agents cooperating in a heterogeneous and dynamic environment. Different tools and data bases as well as inference rules simulating biologists' reasoning can be specified according to the purposes of each project. In ncRNA-Agents,

Table 2. Some ncRNAs identified in the three chromosomes of *S. pombe* by ncRNA-Agents. Column "Sequence Name" shows the chromosome (I, II, III) and column "Genome Position" presents the sequence location (intron, exon, ncRNA exon, and the related gene as annotated in the .gtf file).

Sequence Name	Region Name	Initial Position	Final Position	ncRNA Length (bp)	Genome Position	Strand
Sp_ncRNA_I1	region90	98789	98839	50	Intron Gene: SPAC1F8.05	+
Sp_ncRNA_I2	region553	365148	365236	88	tRNA exon Gene: SPATRNAPRO.01	+
Sp_ncRNA_I3	region707	431217	431301	84	ncRNA Exon Gene: SPNCRNA.645	+
Sp_ncRNA_I4	region1059	617512	617563	51	Intron: After Gene: SPAC1F3.02c	+
Sp_ncRNA_I5	region2501	1421144	1421233	89	Exon Gene: SPAC20G8.10c	+
Sp_ncRNA_II1	region488	355913	356005	92	ncRNA exon Gene: SPNCRNA.1343	+
Sp_ncRNA_II2	region925	570150	570239	89	rRNA Exon Gene: SPBRRNA.30	+
Sp_ncRNA_II3	region2686	1599696	1599799	103	tRNA Exon Gene: SPBTRNAGLY.07	+
Sp_ncRNA_II4	region2914	1774859	1774940	81	Exon Gene: SPBC18H10.03	+
Sp_ncRNA_II5	region6626	3944593	3944692	99	snoRNA Exon Gene: SPBC26H8.02c	+
Sp_ncRNA_III1	region840	489561	489616	55	Intron: After Gene: SPCC970.11c	+
Sp_ncRNA_III2	region1001	583418	583614	196	ncRNA Exon Gene: SPNCRNA.467	+
Sp_ncRNA_III3	region1891	1071041	1071190	149	tRNA Exon Gene: SPCTRNALEU.11	+
Sp_ncRNA_III4	region1910	1102800	1102922	122	tRNA Exon Gene: SPCTRNAGLU.10	+
Sp_ncRNA_III5	region3149	1863119	1863215	96	Exon: After Gene: SPCC1223.10c	+

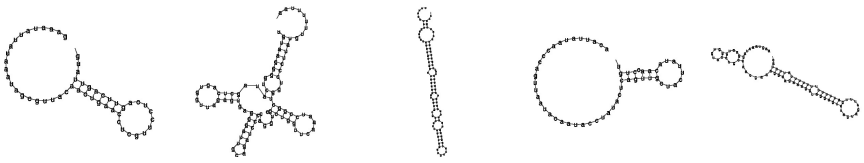


Figure 2. RNAfold analysis of (a) Sp_ncRNA_I4 (C/D Box), (b) Sp_ncRNA_I5 (tRNA), (c) Sp_ncRNA_II5 (H/ACA Box), (d) Sp_ncRNA_III5 (C/D Box), and (e) region 708 (annotated as "snRNA exon gene SPSNRNA.04" by seeker manager).

agents are specialized in different tasks, and can act independently using distinct inference rules. A prototype was implemented using JADE framework and Drools as inference engine for reasoning rules in a web environment. To validate ncRNA-Agents, we executed two experiments to detect ncRNAs in two fungi, *Paracoccidioides brasilienses* and *Schizosacharomyces pombe*. We obtained novel potential ncRNAs for both fungi, showing the usefulness of our approach.

We believe that ncRNA-Agents can strongly help to improve the quality of ncRNA annotation process, considering that massively parallel automatic sequencers produce billions of small fragments. As far as we know, there is no system simulating the biologists reasoning with the integration of different tools and data bases through a multiagent approach as we presented.

Next steps include to improve the expertise of the agents in ncRNA-Agents, using more specific reasoning mechanisms. The inference rules could also be extended, producing a more intensive knowledge based tool. In this direction, we would like to identify, if the annotations are not conclusive enough, a set of new tools to support annotation, e.g., if a potential RNA belongs to a known ncRNA family using a multiple alignment with known and conserved RNAs in related organisms, and figures showing a putative ncRNA folding. Data mining methods could also improve precision in the annotation process. In the collaborative layer, other analyst agents (corresponding to other tools) can be easily included in each manager agent. Particularly, in the class prediction manager, a tool to classify ncRNAs could be included, besides Portrait that only indicate if a sequence is a potential ncRNA. Besides, implementation aspects should be improved allowing a reproducible and portable annotation tool that may be integrated to different annotation systems. Distributed implementation of the agents (besides threads) could reduce time execution, improving the use of ncRNA-Agents in large-scale sequencing projects.

References

1. S. F. Altschul et al. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.
2. R.T. Arrial, R.C. Togawa, and M.M. Brigido. Screening non-coding RNAs in transcriptomes from neglected species using PORTRAIT: case study of the pathogenic fungus *Paracoccidioides brasiliensis*. *BMC Bioinformatics*, 10(1):239, 2009.
3. F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa. JADE - a white paper. *White Paper 3, TILAB - Telecom Italia Lab. (http://jade.tilab.com/)*, v. 3:p. 6–19, 2003.
4. P. Browne. *JBoss Drools Business Rules*. Packt Publishing, 2009.
5. S.R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Research*, 22(11):2079–2088, 1994.
6. S.R. Eddy et al. Non-coding RNA genes and the modern RNA world. *Nature Reviews Genetics*, 2(12):919–929, 2001. Infernal's user guide at http://infernal.janelia.org.
7. M. Fasold et al. DARIO: a ncRNA detection and analysis tool for next-generation sequencing experiments. *Nucleic Acids Research*, doi: 10.1093/nar/gkr357:1304–1351, 2011.
8. M.S.S. Felipe et al. Transcriptional profiles of the human pathogenic fungus *Paracoccidioides brasiliensis* in mycelium and yeast cells. *Journal of Biological Chemistry*, 280(26):24706–24714, 2005.
9. C.L. Forgy. RETE: A fast algorithm for the many pattern/many object pattern match problem. *Artificial intelligence*, 19(1):17–37, 1982.
10. S. Griffiths-Jones. Annotating noncoding RNA genes. *Annu. Rev. Genomics Hum. Genet.*, 8:279–298, 2007.

11. S. Griffiths-Jones et al. Rfam: an RNA family database. *Nucleic Acids Research*, 31(1):439–441, 2003. Rfam database - ftp://ftp.sanger.ac.uk/pub/databases/Rfam.
12. S. Griffiths-Jones et al. miRBase: microRNA sequences, targets and gene nomenclature. *Nucleic Acids Research*, 34:D140–D144, 2006. miRBase database: http://microrna.sanger.ac.uk/.
13. I.L. Hofacker et al. Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie/Chemical Monthly*, 125(2):167–188, 1994.
14. I.L. Hofacker, M. Fekete, and P.F. Stadler. Secondary structure prediction for aligned RNA sequences. *Journal of Molecular Biology*, 319(5):1059–1066, 2002.
15. S. Hoffmann et al. Fast mapping of short sequences with mismatches, insertions and deletions using index structures. *PLoS Comput. Biology*, 5(9):e1000502, 2009.
16. K. Lagesen et al. RNAmmer: consistent and rapid annotation of ribosomal RNA genes. *Nucleic Acids Research*, 35(9):3100–3108, 2007.
17. L. Lestrade and M.J. Weber. snoRNA-LBME-db, a comprehensive database of human H/ACA and C/D box snoRNAs. *Nucleic Acids Research*, 34(suppl 1):D158–D162, 2006.
18. C. Liu et al. NONCODE: an integrated knowledge database of non-coding RNAs. *Nucleic Acids Research*, 33(suppl 1):D112–D115, 2005. NONCODE - http://www.noncode.org.
19. T.M. Lowe and S.R. Eddy. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Ac. Res.*, 25(5):0955–964, 1997.
20. S. Marguerat et al. Quantitative analysis of fission yeast transcriptomes and proteomes in proliferating and quiescent cells. *Cell*, 151(3):671–683, 2012.
21. J.S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29(6-7):1105–1119, 1990.
22. P. Michalak. RNA world–the dark matter of evolutionary genomics. *Journal of Evolutionary Biology*, 19(6):1768–1774, 2006.
23. T. Mituyama et al. The functional RNA database 3.0: databases to support mining and annotation of functional RNAs. *Nucleic acids research*, 37(suppl 1):D89–D92, 2009.
24. K.C. Pang et al. RNAdb 2.0: an expanded database of mammalian non-coding RNAs. *Nucleic acids research*, 35(suppl 1):D178–D182, 2007. RNAdb - http://jsm-research.imb.uq.edu.au/rnadb.
25. K.C. Pang, M.C. Frith, and J.S. Mattick. Rapid evolution of noncoding RNAs: lack of conservation does not mean lack of function. *Trends in Genetics*, 22(1):1–5, 2006.
26. C.G. Ralha, H.W. Schneider, M.E.M.T. Walter, and A.L.C. Bazzan. Reinforcement learning method for BioAgents. In *11th Brazilian Symposium on Neural Networks, Sao Paulo, Brazil*, pages 109–114. IEEE, 2010.
27. C.G. Ralha, H.W. Schneider, M.E.M.T. Walter, and M.M. Brigido. A multi-agent tool to annotate biological sequences. In J. Filipe and A. L. N. Fred, editors, *3rd ICAART 2011, Vol. 2 - Agents. Rome, Italy*, pages 226–231. SciTePress, 2011.
28. M.A. Rosenblad, N. Larsen, T. Samuelsson, and C. Zwieb. Kinship in the SRP RNA family. *RNA Biol.*, 6:508–516, 2009.
29. S.J. Russell and P. Norvig. *Artificial intelligence: A Modern Approach*. Prentice Hall, third edition, 2010.
30. G. Soldà et al. An Ariadne's thread to the identification and annotation of non-coding RNAs in eukaryotes. *Briefings in Bioinformatics*, 10(5):475–489, 2009.
31. M.J. Wooldridge. *An introduction to multiagent systems*. Wiley, 2009.

Apêndice C

Artigo JBCB 2015

Artigo aceito no *Journal of Bioinformatics and Computational Biology* - JBCB, estrato B1 em Periódico no Qualis.

Journal of Bioinformatics and Computational Biology

KNOWLEDGE BASED REASONING TO ANNOTATE NON-CODING RNA USING MULTI-AGENT SYSTEM

Wosley C. Arruda*[1], Daniel S. Souza[1], Célia G. Ralha, Maria Emilia M. T. Walter

Department of Computer Science, University of Brasília, Brazil
warruda@gmail.com, dssouzadan@gmail.com, ghedini@cic.unb.br, mariaemilia@unb.br
[1] *These authors contributed equally to this work*

Tainá Raiol†

Leônidas and Maria Deane Research Center (Fiocruz Amazônia)
tainaraiol@amazonia.fiocruz.br

Marcelo M. Brigido‡

Department of Cellular Biology, Institute of Biology, University of Brasília
brigido@unb.br

Peter F. Stadler§

Department of Computer Science and the Interdisciplinary Center for Bioinformatics, University of Leipzig
peter.stadler@bioinf.uni-leipzig.de

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

Non-coding RNAs (ncRNAs) have been focus of intense research over the last few years. Since characteristics and signals of ncRNAs are not entirely known, researchers use different computational tools together with their biological knowledge to predict putative ncRNAs. In this context, this work presents ncRNA-Agents, a multi-agent system to annotate ncRNAs based on the output of different tools, using inference rules to simulate biologists' reasoning. Experiments with data of the *Saccharomyces cerevisiae* fungus allowed to measure the performance of ncRNA-Agents, with better sensibility, when compared to Infernal, a widely used tool for annotating ncRNA. Besides, data of the *Schizosaccharomyces pombe* and *Paracoccidioides brasiliensis* fungi identified novel putative ncRNAs, which shows the usefulness of our approach. NcRNA-Agents can be be found at: http://www.biomol.unb.br/ncrna-agents.

Keywords: non-coding RNA annotation, knowledge based reasoning, multi-agent system,

*Campus Universitário Darcy Ribeiro, Prédio CIC/EST, Asa Norte, Brasília-DF, CEP: 70910-900
†Rua Teresina, 476, Adrianópolis, Manaus-AM, CEP: 69027-070
‡Campus Universitário Darcy Ribeiro, Prédio do Instituto de Biologia, Asa Norte, Brasília-DF, CEP: 70910-900
§Härtelstrasse 16-18, D-04107, Leipzig, Germany

1. Introduction

Since 1990, important roles of the RNA molecules have been identified, besides the well defined functions of messenger RNA (mRNA), transporter RNA (tRNA) and ribosomal RNA (rRNA), all involved in protein synthesis. Non-coding RNAs (ncRNAs), which are not translated into proteins[6,25], present specific spatial conformation that allow them to play regulatory roles in an extensive variety of biological reactions and processes, e.g., translation initiation, level control of mRNA, stem-cell maintenance, brain developing, metabolism regulation, support to protein transport, nucleotide edition, imprinting regulation and chromatin dynamics[22].

Although worldwide efforts, biological and computational methods are not yet capable to easily identify and classify ncRNAs, directly affecting the annotation of these transcripts. From a biological point of view, ncRNAs are characterized by its transcription and absence of translation into proteins, and RNAs presenting very different nucleotide sequences (primary sequences), but similar spatial conformations (secondary structure) perform the same cellular functions. Therefore, ncRNAs need to be characterized by their secondary structures and not only by their primary sequences. From a computational point of view, ncRNAs can not be identified and classified by homology tools, extensively and efficiently used to annotate protein coding genes (e.g., BLAST[1]), and methods designed to predict secondary from primary structure are commonly used to annotate ncRNAs (Vienna package[13] and Infernal/Rfam[6]). In this context, biologists use different tools, together with their knowledge, to annotate sequences that seem to be ncRNAs.

To cope with this knowledge based reasoning process, we propose to use a multi-agent system (MAS). A MAS is characterized by the distribution of intelligence among autonomous entities called agents, which interact to reach individual or collective goals. In order to do that, the agents in a MAS have to negotiate in a cooperative way, coordinating and joining efforts to reach the objectives, normally not accomplished by a single agent[31].

In this context, this work aims to present ncRNA-Agents, a MAS to support ncRNA annotation, using a knowledge based approach to simulate biologists' reasoning, in order to combine the output of different tools and data bases[28].

2. Challenges to annotate ncRNAs

Methods to identify and classify ncRNAs use different strategies, based on ncRNA characteristics supported by *in silico* and experimental findings. Some criteria are commonly used: a number of known ncRNA classes do not present long ORFs; their sequences have unexpected stop codons; RNAs usually present conservation in their secondary (spatial) structure and rarely in their primary (DNA sequences) structure; some known ncRNAs have complex tridimensional structures, and catalyst or structural functions. These characteristics prevent the use of traditional similarity

based tools to predict proteins[25]. Methods using biological hints are also extensively used to predict ncRNAs: codons, synonymous and non-synonymous substitutions, and minimum folding energy[21].

One important problem when predicting ncRNAs is the lack of available experimental information in order to confirm *in silico* prediction. Although computational predictions are useful, confirmation needs biological experiments, such as RNA-seq of small RNAs, real-time PCR and gene deletion or knockout experiments. But the absence of translation is not conclusive, since predicted ncRNA may not be transcribed or translated when some environmental or physiological conditions happen. Therefore, to predict ncRNAs, biologists first use different bioinformatics tools, with distinct methods, and after make a combined analysis of all these information to decide if a sequence is a potential ncRNA. We have three main problems when annotating ncRNAs: secondary structure prediction[14], secondary structure comparison[13,6] and ncRNA identification and classification[10].

2.1. *Tools and data bases*

In this section, some commonly used computational tools and databases[30] to identify and classify ncRNAs are briefly described. Then, we propose a way to group these tools, together with a reasoning process to infer annotation, combining and evaluating these tools' results.

BLAST (*Basic Local Alignment Search Tool*)[1] is extensively used to predict proteins. Although Blast can successfully detect some specific ncRNA families (e.g., snoRNAs), usually, only nucleotide comparisons are not sufficient, since many ncRNA classes are very different regarding to the primary sequences, but exhibit a common secondary structure.

Infernal (INFERence of RNA Alignment)[5,6] identifies ncRNAs looking for secondary structures. It builds profiles of RNA consensus spatial structures, known as covariance models (CMs), in order to find similarities between the investigated sequence and the consensus secondary structure of each one of the RNA families stored in the Rfam database[11].

tRNAscan[19] is considered one of the most accurate tRNA predictors. It combines three programs: two tRNA predictors and a covariance model[5] previously trained with tRNA sequences. The execution of these three programs results is a tRNA identifier presenting high sensibility ($99 - 100\%$) and specificity (with a false positive rate less than 0.00007 by Mb) in a reasonable velocity.

Portrait[2] identifies ncRNAs of not complete transcriptomes, of yet not entirely characterized species, based on Support Vector Machine (SVM). Portrait's result is a probability indicating the likelihood of a transcript to be non-protein coding.

Vienna[13] is a set of packages used to generate or to compare RNA secondary structures. Folding in this tool uses prediction algorithms based on RNA free energy and the probability of base pairing[21]. Particularly, RNAfold[14] package is based on the hypothesis that an RNA molecule is folded in a more stable thermodynamic

structure with the minimum free energy. RNAz[32] is based in minimum free energy (MFE) to predict an RNA two-dimensional structure.

Considering commonly used ncRNA data bases, created from experimental and computational data[30], the following may be cited.

NONCODE[18] includes many different types of ncRNAs, except tRNAs and rRNAs. More than 80% of the input in NONCODE are experimentally confirmed. We used NONCODE version v3.0, with more than 411,552 ncRNAs. **RNAdb**[24] contains sequences and annotation of thousands of mammalian ncRNAs, but a large number of them do not have known biological functions. **miRBase**[12] is a data base of microRNAs, providing a searchable data base of previously published miRNA sequences and annotation, and also miRNA gene hunters for novel miRNA genes, not yet published. **snoRNA** database[17] contains human snoRNAs of both types, H/ACA box and C/D box. **fRNAdb**[23] integrates a set of data bases, including NONCODE and RNAdb. **Rfam**[11] is a curated data base containing information about ncRNA families, presenting two classes of data: CM profile and seed alignments for each ncRNA family. In this work, we used Rfam 11.0, with 2,208 families.

2.2. *Grouping ncRNA annotation tools*

We propose to group computational tools to annotate ncRNAs as follows:

- **Homology**: ncRNAs are predicted using alignment among sequences. These comparisons depend on the quality of the annotation stored in data bases, since this quality affects ncRNA classification. BLAST[1] and Infernal[6] are two widely used tools, both using the concept of similarity (the greater the similarity the greater the chance that both sequences have conserved the same function). Generally speaking, BLAST does not produce good results to annotate ncRNAs, in contrast to Infernal, which is more sensitive and specific to classify hundreds of different types of ncRNAs;
- **Class prediction**: ncRNAs are identified using methods based on machine learning. In the supervised learning paradigm, we take a set of known ncRNAs and a set of known proteins, computing *ab initio* characteristics for these sequences, in order to create a model to predict ncRNAs. Examples are PORTRAIT[2] and DARIO[7]; and
- ***De novo* model**: ncRNAs are predicted using models distinct from *homology* and *class prediction*, e.g., the Vienna[13] packages based on thermodynamic models.

2.3. *Knowledge based reasoning process*

In order to annotate a sequence, human annotators combine and evaluate results of computational tools using their biological knowledge. This reasoning process can be modelled as follows.

First, since results of different tools provide information about different aspects of a sequence (the motivation to group computational tools), analyzing them all

together refines annotation. Thus, results from tools of the homology group give characteristics regarding to sequence homology: tRNAscan annotate if a sequence is a tRNA; Blast shows good alignments relative to adopted ncRNA data bases, and can indicate the same annotation as tRNAscan (i.e., a tRNA), or suggests a different one; Infernal can recommend tRNA, as tRNAscan and/or Blast, or yet gives other suggestions. Then, the human annotator, looking for characteristics and measures computed by each of these tools, can decide the annotation to the sequence. Following, results from tools of the class prediction group, e.g., Portrait (computes the probability of sequence be a ncRNA) can confirm the suggestion inferred by the homology group tools, or not. Yet, another tool, e.g., RNAfold from Vienna package, shows if the investigated sequence can be folded with stable conformational energy.

If the tools of each group suggest the same annotation, the recommended annotation for the sequence is considered "good". Otherwise, if the tools suggest annotations that partially agree, or do not agree at all, or if only some tools recommend annotation, these results have to be analyzed and combined, considering their reliability to annotate ncRNAs, which depend on each project. In this case, the human annotator can evaluate if the inferred annotation is "reliable", verifying (e.g., if information about transcripts and genome of the same/very close organism is available) if the sequence is transcribed, and if it is conserved among related organisms. A predicted ncRNA presenting its coding region conserved among related organisms reinforces annotation reliability. Moreover, if the adopted tools do not suggest any "good" or "reliable" annotation, seeking for similar sequences annotated as ncRNAs in related organisms suggests a "novel ncRNA" for the investigated sequence.

3. MAS and inference rules

In this section, we briefly describe concepts related to MAS, as well as inference rules, respectively used for knowledge representation and reasoning. First, we present the concept of an agent[29], an entity that interacts with the environment, perceiving it through sensors and acting using actuators. An agent has two important characteristics: it is capable to act autonomously, taking decisions leading to the satisfaction of its objectives; and it can interact with other agents through human inspired social interaction protocols. We may cite as agents functionalities: coordination, cooperation, competition and negotiation.

A MAS includes many homogeneous or heterogeneous agents interacting and working together. Each agent operates asynchronously related to other agents[31]. In order to have agent communication and interaction we need to have an adequate MAS infrastructure with specific agent language and interaction protocol.

JADE (Java Agent DEvelopment Framework)[3] is a commonly used framework to develop MAS, since it follows the patterns of FIPA (Foundation for Intelligent Physical Agents), an international organization responsible to define patterns for the development of agent technologies. JADE allows and facilitates the development

of agents using Java language since it has many already defined functions, uses ACL (Agent Communication Language), and has ready-to-use interaction protocols (e.g., contract net). JADE presents a visual interface to monitor agents' execution that helps to control these agents' life cycle, having also many built-in resources, e.g., the directory facilitator (yellow pages) and the agent controller manager (white pages). In this work, we have used JADE to implement the ncRNA-Agents prototype. *JADE* is a free software distributed under *LGPL* license, allowing portability due to the *Java* language.

Reasoning agent skills can be implemented using a knowledge based approach with rules and an inference engine. Drools[4] is an open source inference engine that allows to build a knowledge base using forward and backward chaining methods, based on the RETE algorithm[9]. In this work, we used Drools to implement agents' reasoning skills to simulate biologists' knowledge and reasoning inference method.

4. ncRNA-Agents

In this section, we describe the ncRNA-Agents architecture and prototype, which are inspired in previous works of our team[26,27,28].

NcRNA-Agents architecture has four different layers, as presented in Figure 1:

- **interface layer**: receives a sequence, or a file of sequences, in FASTA format, together with the annotation tools chosen by a user. The system will return, for each sequence, the putative ncRNA annotation (if it is the case), along with the results computed by the tools, which were used in the reasoning process;
- **conflict resolution layer**: receives the user request and, simulating a biologist's reasoning to analyze the suggestions received from the collaborative layer, recommends the annotation to be sent to the interface layer;
- **collaborative layer**: executes the tools (chosen by the user) to annotate ncRNAs, filtering the results to obtain useful data, and sends these filtered results to the conflict resolution layer; and
- **physical layer**: acesses and stores different data bases (Section 2.1).

Note that the collaborative layer presents different levels of agents (Figure 1): manager and analyst. Manager agents are responsible to filter the suggestions sent by the analyst agents. There are five types of manager agents, three of them defined according to the tools' grouping proposed in Section 2.2, and two created to evaluate the recommended annotation (see Section 2.3), i.e., removing false positives, or looking for novel ncRNAs by investigating sequence conservation in related organisms. The manager agents also simulate the reasoning of the biologists:

- **homology manager**: coordinates agents working with homology tools, e.g., tRNAscan[19], BLAST[1] and Infernal[5,6];
- **class prediction manager**: coordinates agents working with machine learning tools, e.g., Portrait[2];

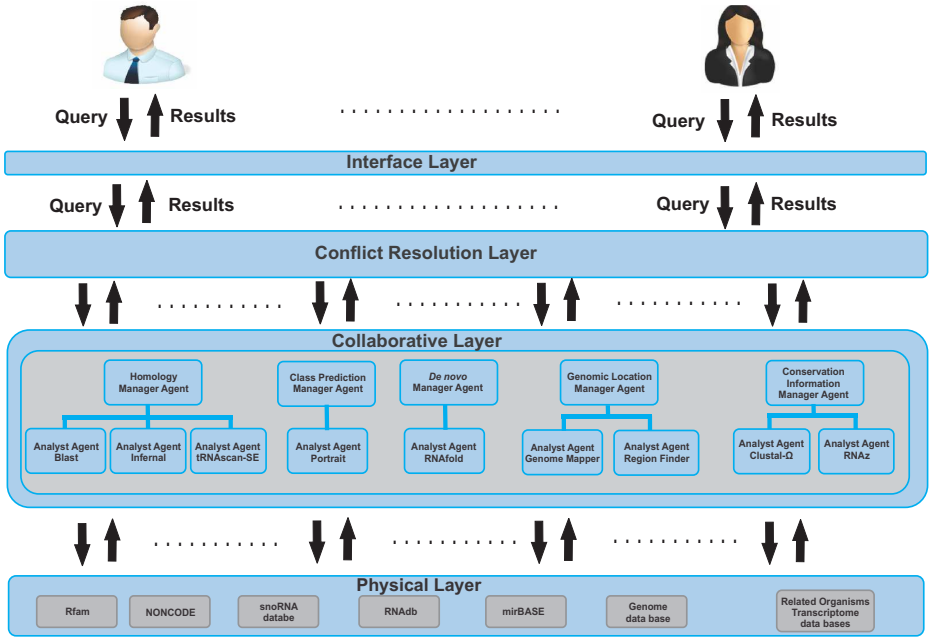


Fig. 1. The ncRNA-Agents architecture.

- ***de novo* manager**: coordinates agents working on tools that do not use reference organisms, e.g., RNAfold[14] from Vienna package;
- **genomic location manager**: coordinates agents using tools to locate a predicted ncRNA: (i) in a reference genome G, if the sequence is mapped to G; or (ii) in exons, introns or intergenic regions, if this information is available; and
- **conservation information manager**: coordinates agents executing tools for seeking conservation among related organisms.

Finally, analyst agents are responsible to execute specific tools to annotate ncRNAs. Each analyst agent (created from a manager agent request) parses the output file generated by its corresponding tool. This analysis is returned as an annotation to its manager agent. Particularly, analyst agents were created in the genomic location manager to improve annotation, e.g., to verify if a ncRNA candidate is located in an intronic, exonic or intergenic region, or the location in a reference genome. The conservation information manager investigates conserved domains among related organisms, using two analysts, one to perform multiple alignment (e.g., using Clustal-Ω), which is the input to another agent that computes the probability of the conserved spatial structures are RNAs (e.g., using RNAz).

A prototype was created as presented in Figure 1, and can be accessed at http://www.biomol.unb.br/ncrna-agents. The ncRNA-Agents system was imple-

mented with JADE version 4.3.3[3]. Besides, ncRNA-Agents used *Drools* version 6.1.0.Final for the agents' reasoning[4]. We formalized biological knowledge using declarative rules, according to parameters adopted for each project. As an example, Figure 2 shows the conflict resolution layer (CRL) rules, which are based on the results sent by the managers of the collaborative layer.

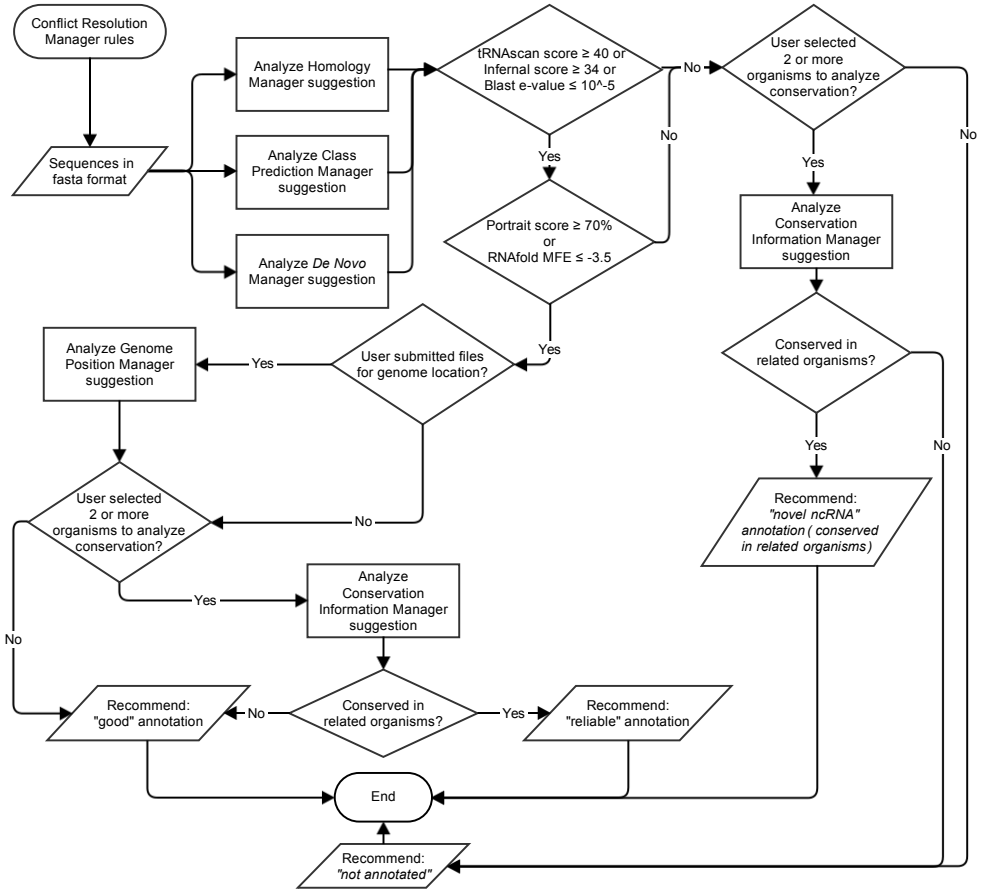


Fig. 2. Rules used in the Conflict Resolution Manager to simulate biologists' annotation reasoning.

Rules for the homology manager were created for choosing, among the results of the three analysts (Blast, Infernal and tRNAscan), the best annotation, which is sent to CRL. TRNAscan analyst verifies if there are good alignments, selecting the alignment with the highest score. Blast analyst verifies if there are alignments with $e-value \leq 10^{-5}$ (this threshold is an easily changeable parameter). Infernal analyst verifies if there are alignments with score ≥ 34. To recommend annotation, the order chosen by the homology manager is Infernal, tRNAScan, and Blast, in this order,

for the tools that produced results, according to the thresholds adopted for each tool. We note that these analyst agents execute in parallel, using memory threads to increase efficiency. Class prediction manager sends to CRL the probability of being ncRNA, as computed by Portrait analyst, while *de novo* manager sends to CRL the folding energy calculated by RNAfold analyst. The other analysts, belonging to the genomic location and conservation managers, are used as described in Section 2.3.

The web server was implemented using Apache Tomcat 7 and JSF version 2.2.6.

5. Results

The performed experiments included three case studies. First, we compared known annotated ncRNAs of the *Saccharomyces cerevisiae* fungus with the annotation recommended by ncRNA-Agents to measure the performance of the prototype. Afterwards, we identified novel ncRNAs in the *Paracoccidioides brasilensis* and *Schizosaccharomyces pombe* fungi.

5.1. *NcRNA-Agents performance*

To measure the performance of ncRNA-Agents prototype, we used a positive set of 417 known ncRNAs of the S288C assembly, comprising 77 snoRNAs, 299 tRNAs, 6 snRNAs, 1 telomerase RNA, 16 rRNAs and 18 predicted ncRNAs, available at http://www.yeastgenome.org/. For the negative set, we selected 417 protein coding transcripts of the RM11-1a assembly, available at http://www.broadinstitute.org/. The protein transcripts were chosen based on the following criteria: size greater than or equal to 30 nucleotides and less than or equal to $1,500$ nucleotides; each transcript has to be aligned to a sequence stored in SwissProt or NR database with e-value $\leq 10^{-10}$, and this alignment has to have a valid functional annotation (different from "hypothetical protein" and related terms), confirmed with the annotation description of each family domain, retrieved from ProDom database, with a score greater than or equal to 80%.

The prototype was executed with all agents shown in Figure 1. According to Broad Institute, the sequence divergence between assemblies RM11-1a and S288C is estimated in $0.5\% - 1\%$, which justifies the use of the RM11-1a assembly as the reference genome, and its corresponding $.gtf$ file containing the location of genes (start and stop regions, and exons), all available at http://www.broadinstitute.org/, to assess the location of the transcripts in the positive and negative sets, i.e., to find if a transcript is in an intergenic, an exonic, or an intronic region. A transcript location is found based in an overlap coefficient OC, such that if $OC \geq 80\%$ (meaning that the transcript has an overlap with a genome region of at least 80%), then the location of the transcript can be estimated in the reference genome, and its region (intronic, exonic or intergenic) can be identified.

To set the best values for the parameters of tRNAScan and Portrait, we investigated sensitivity, specificity and accuracy for different parameters of tRNAScan

and Portrait. These empirical measures indicated that Portrait does not affect sensitivity, unlike tRNAscan, and the best tRNAScan score is 30, which was used for all the case studies.

Table *Sc_comparison_table.xls*, in the supplementary material, shows a comparison between the annotation of ncRNA-Agents and Infernal, for all 417 known ncRNAs (positive set). In this table, from the 417 known ncRNAs, ncRNA-Agents identified 402 as ncRNAs (15 were not annotated as ncRNAs) while Infernal identified 400 as ncRNAs (17 were not annotated as ncRNAs). In fact, both methods annotated exactly the same functions, but two *mRNAlike lncRNAs* annotated by ncRNA-Agents were not annotated by Infernal.

Regarding to the 417 proteins of the negative set, Infernal did not annotate any of them, while ncRNA-Agents annotated 24 as novel ncRNAs (393 were not predicted as ncRNAs). We note that, from these 24 novel ncRNAs, 10 were previously annotated as ribosomal RNAs and 2 as hypothetical proteins, suggesting an annotation different from the previous ones. Therefore, our system could be used to improve a large number of innacurate annotations stored in databases.

Therefore, we obtained the following measures of ncRNA-Agents and Infernal respectively: $sensitivity = 96.40\%$ and 95.92%; $specificity = 94.24\%$ and 100%; and $accuracy = 95.32\%$ and 97.96%.

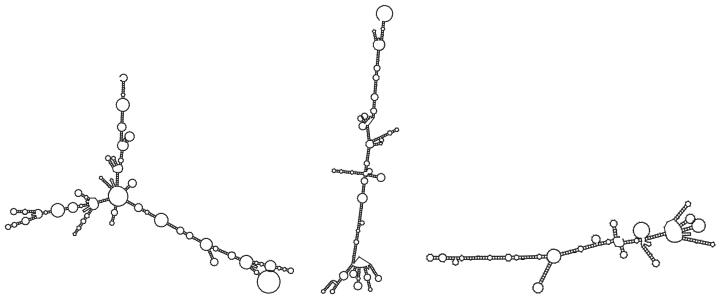
5.2. *Novel ncRNAs in P. brasiliensis*

To validate ncRNA-Agents for annotating novel ncRNAs, we developed an experiment with the fungus *Paracoccidioides brasiliensis* (*Paracoccidioides lutzii* Pb01). Data from $8,826$ transcripts and 110 supercontigs, both from *P. lutzii* Pb01 were downloaded from http://www.broadinstitute.org/.

Mapping of transcripts was performed with Segehmel[15], taking as reference these supercontigs. Next, a Perl script extracted, based in the *.sam* file generated by mapping, the DNA sequence of the mapped location, in the supercontig sequences, for each of the $4,502$ mapped transcripts. These $4,502$ transcript DNA sequences were submitted to Blast with the SwissProt data base, with an e-value of 10^{-5}, which annotated $2,517$ proteins. Next, $1,985$ not annotated proteins were submited to UniprotKB, which annotated $1,969$ sequences. Both sets of 16 not annotated ("no hits") sequences and of 429 annotated as "hypothetical protein" and related terms (not only the best hit, but all the hits above the threshold), a total of 445 sequences, were submitted as input to ncRNA-Agents.

The prototype used all the managers shown in Figure 1. Table 1 (Appendix A) shows the only three annotated novel ncRNAs in *P. brasiliensis*, noting that these ncRNAs were conserved in some of the related fungi *Blastomyces dermatitidis* er 3.1, *Coccidioides immitis* rs finished, *Histoplasma capsulatum* h88, *Aspergillus fumigatus*, *Aspergillus nidulans* and *Aspergillus terreus*, all of them downloaded from http://www.broadinstitute.org/. Details are shown in the website http://www.biomol.unb.br/ncrna-agents.

Figure 3 shows the ncRNAs in *P. brasiliensis* identified by ncRNA-Agents. It is noteworthy that the ncRNAs predicted by ncRNA-Agents did not have any common sequences when compared to the ones predicted only by Portrait[2].



(a) SC 2 (b) SC 4 (c) SC 8

Fig. 3. Spatial structures of the ncRNAs of Table 1, where SC means supercontig.

5.3. *NcRNA annotation in S. pombe*

Another experiment was developed with the *Schizosacharomyces pombe*[20] fungus. For this case study we used information of *S. pombe* genome and transcriptome.

Small RNA-Seq data of *S. pombe* was extracted from EMBL-EBI (experiment E-MTAB-1154). After filtering to remove low quality reads, mapping was performed with Segehmel[15], taking as reference the *S. pombe* genome (downloaded from http://www.broadinstitute.org/). Next, a Perl script extracted the DNA genomic regions presenting at least five mapped reads, and another Perl script found the location of each region in the *.sam* file generated by mapping. These mapped regions were submitted to Blast with SwissProt data base, to annotate proteins. Sequences not identified as proteins ("no hits"), together with sequences annotated as "hypothetical protein", "predicted protein", "potential protein" and "unknown sequence", were submitted as input to ncRNA-Agents.

Homology, class prediction and *de novo* managers were executed, as in in the previous experiment. In the genomic location manager, genome finder analyst agent verified if the predicted ncRNA was located in an exon, an intron or an intergenic region, since we had the *.gtf* file. An expressed sequence was considered to be in an intronic region if it was located inside a gene presenting an intersection of at most 70% with an exon.

In the three chromosomes of the fungus *S. pombe*, 9,566, 7,565 and 4,126 DNA genomic regions were found, respectively. From these, a Blast against SwissProt found 1,795, 1,562 and 802 sequences, in chromosomes 1, 2 and 3, annotated as proteins. Thus, a total of 17,098 sequences (7,771 from chromosome 1, 6,003 from

chromosome 2, and 3,324 from chromosome 3) were submitted as input to ncRNA-Agentes, which detected 40 novel ncRNAs (21 conserved and 19 not conserved in related organisms), all in the "+" strand, having both strands been verified. General characteristics of these ncRNAs are shown in Tables 2 and 3, with spatial structure shown in Figures 4 and 5.

Note that, in both cases (predicted ncRNAs obtained from all the managers and those by some of the managers), to be predicted as novel ncRNAs, a sequence had to present folding free energy < -3.50 kcal/mol (according to RNAfold and RNAz), indicating likely stable conformational structures.

Conservation for the *S. pombe* sequences were investigated in the following organisms (the closer organisms with available information): *Pneumocystis murina*, *Schizosaccharomyces cryophilus*, *Schizosaccharomyces japonicus* and *Schizosaccharomyces Octosporus*, all of them downloaded from http://www.broadinstitute.org/.

Sequences not annotated as ncRNAs by homology, class prediction or *de novo* manager were submitted to the genomic location manager (using annotation of the *S. pombe .gtf* file) and conservation information manager, which allowed to find:

- 21 novel ncRNAs, all of them presenting conservation in at least two related organisms: (i) in chromosome I, 6 in exons (3 tRNAs, 3 snoRNA) and 1 snoRNA in intron; (ii) in chromosome II, 8 in exons (5 tRNAs, 1 rRNA, 2 snRNA); and (iii) in cromossomo III, 6 in exons (5 tRNAs, 1 snRNA);
- 19 novel ncRNAs, with no conservation in related organisms: (i) in chromosome I, 9 in exons (8 snoRNAs, 1 lncRNA), and 1 lncRNA, 1 snRNA in intron; (ii) in chromosome II, 4 in exons (1 tRNA, 1 snRNA, 2 snoRNAs) and 2 lncRNAs in introns; and (iii) in cromossomo III, 2 snRNAs in exons.

5.3.1. *Annotated ncRNAs with/without conservation*

Table 2 (Appendix B) shows the ncRNAs of *S. pombe* annotated by ncRNA-Agents, presenting conservation in related fungi, all of them occuring in the strand +.

Figure 4 shows the consensus secondary structures of the ncRNAs among *S. pombe* and related fungi.

Table 3 (Appendix B) shows the ncRNAs of *S. pombe* annotated by ncRNA-Agents, predicted by Portrait and RNAfold, but not presenting conservation in other fungi, noting that all of them occur in the strand +.

Figure 5 shows the consensus secondary structure of the ncRNAs in *S. pombe*, but not found in related fungi.

5.3.2. *Summary of the annotated ncRNAs in S. pombe*

Table 4 shows the total number and types of ncRNAs identified in *S. pombe*, with/without conservation in related organisms.

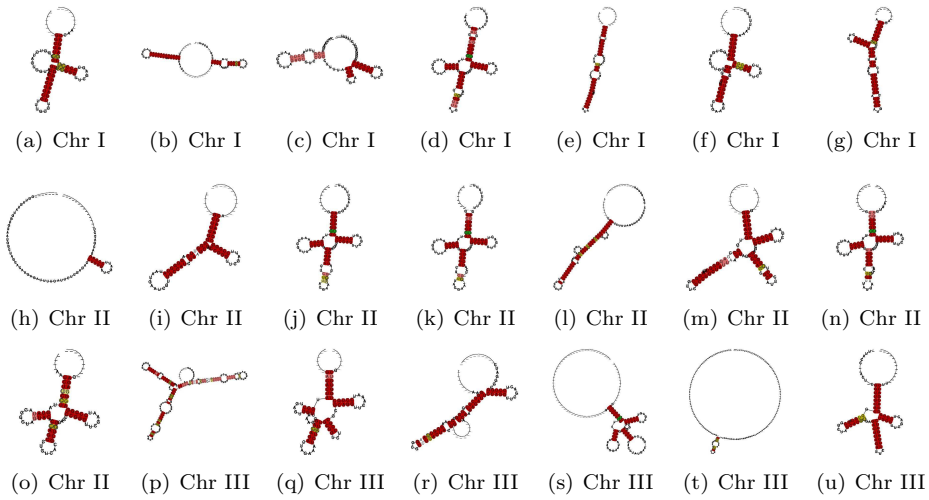


Fig. 4. Spatial structures of the ncRNAs of Table 2, conserved in at least two related fungi.

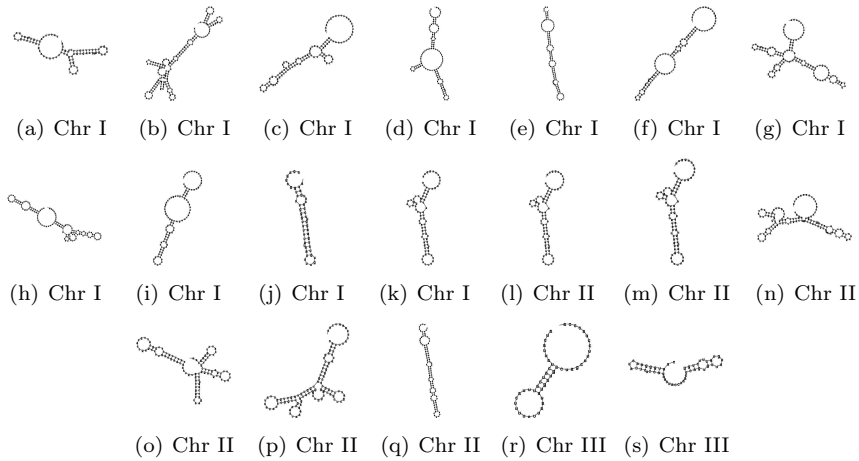


Fig. 5. Spatial structures of the ncRNAs of Table 3, not conserved in related fungi.

6. Conclusion

In this article, we presented ncRNA-Agents, a multi-agent system to recommend ncRNA annotation. The proposed architecture allows the execution of intelligent agents cooperating in a heterogeneous and dynamic environment. Different tools and data bases as well as inference rules simulating biologists' knowledge can be specified according to the purposes of each project. In ncRNA-Agents, agents are specialized in different tasks, and can act independently using distinct inference

rules. A prototype was implemented using JADE framework and Drools as inference engine for reasoning rules in a web environment. To validate the prototype, ncRNA-Agents was executed to detect ncRNAs in *Saccharomyces cerevisiae*, being these results compared to the results obtained with Infernal. The annotated ncRNAs of *S. cerevisiae* were considered the gold standard. In addition, we executed two other experiments to detect ncRNAs in *Schizosacharomyces pombe* and in *Paracoccidioides brasilienses*. For both organisms, we obtained novel ncRNAs only looking for conservation in related organisms.

We believe that ncRNA-Agents can help to improve quality of the ncRNA annotation process, considering that high throughput sequencers produce billions of small fragments. As far as we know, this system is original, regarding to the simulation of biologists' reasoning, with the integration of different tools and data bases through a multi-agent approach as we presented.

Next steps include to improve the expertise of the agents in ncRNA-Agents, using other reasoning mechanisms. Particularly, in the collaborative layer, other analyst agents (corresponding to other tools) could be easily included in each manager agent, e.g., snoStrip[34]. In the class prediction manager, a tool to classify ncRNAs could be included, besides Portrait, which only indicate if a sequence is a potential ncRNA, e.g., snoReport[33]. Data mining methods could be used to enlarge the number and variety of organisms in the physical layer, as well as to discover novel patterns among the available ncRNAs in ncRNA-Agents. Moreover, agents could store knowledge about taken decisions, and use this information to improve future annotations. A distributed implementation could be used to reduce time execution, allowing to use ncRNA-Agents in large-scale sequence projects.

Acknowledgments

WCA and DSS would like to thank CAPES for the Doctorate support. MMB and MEMTW would like to thank CNPq for the research fellowship. All the authors would like to thank Waldeyr M. C. Silva for the support to install the multiagent system and the website in the bioinformatics laboratory of the University of Brasilia, and the referees for their valuable comments.

Appendix A. Annotated ncRNAs in *P. brasiliensis*

Table 1. Novel ncRNAs in *P. brasiliensis* identified by ncRNA-Agents.

Super-contig (v2)	ncRNA-Agents Annotation	Initial Position	Final Position	ncRNA Length (bp)	Strand	Figure label
2	Novel ncRNA	734814	733918	896	-	(a)
4	Novel ncRNA	629991	630682	691	+	(b)
8	Novel ncRNA	353736	354167	431	+	(c)

Appendix B. Annotated ncRNAs in *S. pombe*

Table 2. NcRNAs in *S. pombe* identified by ncRNA-Agents in chromosomes I, II and III presenting conservation in related organisms, all found in strand +.

Chr	ncRNA-Agents Annotation	Initial Position	Final Position	ncRNA Length (bp)	Figure label
I	tRNA	365148	365236	88	(a)
I	snoRNA	1772288	1772406	118	(b)
I	snoRNA	2441955	2442050	95	(c)
I	tRNA	3086302	3086405	103	(d)
I	snoRNA	3700659	3700749	90	(e)
I	tRNA	3701548	3701638	90	(f)
I	snoRNA	3776767	3776848	81	(g)
II	tRNA	355913	356005	92	(h)
II	rRNA	570150	570239	89	(i)
II	tRNA	599696	1599799	103	(j)
II	tRNA	1645034	1645137	103	(k)
II	snRNA	3236865	3236992	127	(l)
II	snRNA	3350583	3350694	111	(m)
II	tRNA	3814123	3814225	102	(n)
II	tRNA	3939221	3939306	85	(o)
III	snR99	583418	583614	196	(p)
III	tRNA	847694	847787	93	(q)
III	tRNA	925843	925933	93	(r)
III	tRNA	1071041	1071190	149	(s)
III	tRNA	1102800	1102922	122	(t)
III	tRNA	1863119	1863215	96	(u)

Table 3. NcRNAs identified *S. pombe* by ncRNA-Agents in chromosomes I, II and III, not presenting conservation in related organisms, all found in strand +.

Chr	ncRNA-Agents Annotation	Initial Position	Final Position	ncRNA Length (bp)	Figure label
I	snoRNA	431217	431301	84	(a)
I	snoRNA	526920	527092	172	(b)
I	snoRNA	527188	527301	113	(c)
I	snoRNA	1200132	1200257	125	(d)
I	snoRNA	1200391	1200508	117	(e)
I	snoRNA	1712992	1713099	107	(f)
I	snoRNA	2416362	2416500	138	(g)
I	snoRNA	3699719	3699850	131	(h)
I	snoRNA	4516055	4516155	100	(i)
I	mRNAlike	4516842	4516892	50	(j)
I	mRNAlike	5021770	5021853	83	(k)
II	mRNAlike	1965085	1965168	83	(l)
II	mRNAlike	1969651	1969734	83	(m)
II	snoRNA	2044199	2044327	128	(n)
II	snRNA	3020253	3020353	100	(o)
II	tRNA	3814551	3814659	108	(p)
II	snoRNA	3944593	3944692	99	(q)
III	snRNA	2419266	2419316	50	(r)
III	snRNA	2419595	2419645	50	(s)

Table 4. Total of ncRNAs of *S. pombe*, with/without conservation in related fungi.

Chromosome	Type					
	tRNA exon	ncRNA exon	rRNA exon	snRNA exon	intron	exon
I	2/0	3/7	0/0	0/0	1/2	1/2
II	2/1	1/1	2/0	1/2	0/2	2/0
III	2/0	1/2	0/0	0/0	0/0	3/0
Total	6/1	5/10	2/0	1/2	1/4	6/2

References

1. S. F. Altschul et al. Basic Local Alignment Search Tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.
2. R.T. Arrial, R.C. Togawa, and M.M. Brigido. Screening non-coding RNAs in transcriptomes from neglected species using PORTRAIT: case study of the pathogenic fungus *Paracoccidioides brasiliensis*. *BMC Bioinformatics*, 10(1):239, 2009.
3. F.L. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-Agent Systems with JADE*. Wiley, 2007.
4. P. Browne. *JBoss Drools Business Rules*. Packt Publishing, 2009.
5. S.R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Research*, 22(11):2079–2088, 1994.
6. S.R. Eddy et al. Non-coding RNA genes and the modern RNA world. *Nature Reviews Genetics*, 2(12):919–929, 2001. Infernal's user guide at http://infernal.janelia.org.
7. M. Fasold et al. DARIO: a ncRNA detection and analysis tool for next-generation sequencing experiments. *Nucleic Acids Research*, doi: 10.1093/nar/gkr357:1304–1351, 2011.
8. M.S.S. Felipe et al. Transcriptional profiles of the human pathogenic fungus *Paracoccidioides brasiliensis* in mycelium and yeast cells. *Journal of Biological Chemistry*, 280(26):24706–24714, 2005.
9. C.L. Forgy. RETE: A fast algorithm for the many pattern/many object pattern match problem. *Artificial intelligence*, 19(1):17–37, 1982.
10. S. Griffiths-Jones. Annotating noncoding RNA genes. *Annu. Rev. Genomics Hum. Genet.*, 8:279–298, 2007.
11. S. Griffiths-Jones et al. Rfam: an RNA family database. *Nucleic Acids Research*, 31(1):439–441, 2003. Rfam database - ftp://ftp.sanger.ac.uk/pub/databases/Rfam.
12. S. Griffiths-Jones et al. miRBase: microRNA sequences, targets and gene nomenclature. *Nucleic Acids Research*, 34:D140–D144, 2006. miRBase database: http://microrna.sanger.ac.uk/.
13. I.L. Hofacker et al. Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie/Chemical Monthly*, 125(2):167–188, 1994.
14. I.L. Hofacker, M. Fekete, and P.F. Stadler. Secondary structure prediction for aligned RNA sequences. *Journal of Molecular Biology*, 319(5):1059–1066, 2002.
15. S. Hoffmann et al. Fast mapping of short sequences with mismatches, insertions and deletions using index structures. *PLoS Comput. Biology*, 5(9):e1000502, 2009.
16. K. Lagesen et al. RNAmmer: consistent and rapid annotation of ribosomal RNA genes. *Nucleic Acids Research*, 35(9):3100–3108, 2007.
17. L. Lestrade and M.J. Weber. snoRNA-LBME-db, a comprehensive database of human H/ACA and C/D box snoRNAs. *Nucleic Acids Research*, 34(suppl 1):D158–D162, 2006.
18. C. Liu et al. NONCODE: an integrated knowledge database of non-coding

RNAs. *Nucleic Acids Research*, 33(suppl 1):D112–D115, 2005. NONCODE - http://www.noncode.org.
19. T.M. Lowe and S.R. Eddy. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Ac. Res.*, 25(5):0955–964, 1997.
20. S. Marguerat et al. Quantitative analysis of fission yeast transcriptomes and proteomes in proliferating and quiescent cells. *Cell*, 151(3):671–683, 2012.
21. J.S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29(6-7):1105–1119, 1990.
22. P. Michalak. RNA world–the dark matter of evolutionary genomics. *Journal of Evolutionary Biology*, 19(6):1768–1774, 2006.
23. T. Mituyama et al. The functional RNA database 3.0: databases to support mining and annotation of functional RNAs. *Nucleic acids research*, 37(suppl 1):D89–D92, 2009.
24. K.C. Pang et al. RNAdb 2.0: an expanded database of mammalian non-coding RNAs. *Nucleic acids research*, 35(suppl 1):D178–D182, 2007. RNAdb - http://jsm-research.imb.uq.edu.au/rnadb.
25. K.C. Pang, M.C. Frith, and J.S. Mattick. Rapid evolution of noncoding RNAs: lack of conservation does not mean lack of function. *Trends in Genetics*, 22(1):1–5, 2006.
26. C.G. Ralha, H.W. Schneider, L.O. Fonseca, M.E.M.T. Walter, and M.M. Brigido. Using Bioagents for supporting manual annotation on genome sequencing projects. In *Proceeding of the 3rd Brazilian Symposium on Bioinformatics - BSB 2008, Brazil*, pages 127–139. Springer-Verlag, Berlin, Heidelberg, 2008.
27. C.G. Ralha, H.W. Schneider, M.E.M.T. Walter, and A.L.C. Bazzan. Reinforcement learning method for BioAgents. In *11th Brazilian Symposium on Neural Networks, Sao Paulo, Brazil*, pages 109–114. IEEE, 2010.
28. C.G. Ralha, H.W. Schneider, M.E.M.T. Walter, and M.M. Brigido. A multi-agent tool to annotate biological sequences. In J. Filipe and A. L. N. Fred, editors, *3rd ICAART 2011, Vol. 2 - Agents. Rome, Italy*, pages 226–231. SciTePress, 2011.
29. S.J. Russell and P. Norvig. *Artificial intelligence: A Modern Approach*. Prentice Hall, third edition, 2010.
30. G. Soldà et al. An Ariadne's thread to the identification and annotation of noncoding RNAs in eukaryotes. *Briefings in Bioinformatics*, 10(5):475–489, 2009.
31. M.J. Wooldridge. *An introduction to multi-agent systems*. Wiley, 2009.
32. M. Zuker and P. Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic acids research*, 9(1):133–148, 1981.
33. J. Hertel and I. Hofacker and P. F. Stadler. SnoReport: computational identification of snoRNAs with unknown targets. *Bioinformatics*, 24:158–164,2008.
34. S. Bartschat and S. Kehr and H. Tafer and P. F. Stadler and J. Hertel. snoStrip: a snoRNA annotation pipeline. *Bioinformatics*, 30(1):115-116, 2014.

Wosley C. Arruda is a PhD student, in bioinformatics, at the Department of Computer Science from University of Brasilia. He received his System Analysis degree from the University Salgado Filho, and his Mastership in Computer Engineering and in Electrical Engineering in 2007 at the Federal University of Goiás, all in Brazil. From 2010 to 2011, he worked in the Organization of Ibero-American States (OEI) from the Ministry of Education.

Daniel S. Souza is a PhD student, in bioinformatics, at the Department of Computer Science from University of Brasilia. He received his Bachelor's degree in Computer Science from the Instituto de Educação Superior de Brasília, and his Mastership in Informatics in 2014 at the University of Brasília, all in Brazil.

Célia G. Ralha received her Ph.D. in Computer Science from Leeds University, UK, in 1996. Since 2002, she is an Associate Professor at the Department of Computer Science, University of Brasília, Brazil. She has has been working on Intelligent Information Systems, with research interest on the treatment of information and knowledge using multi-agent systems approach.

Tainá Raiol received her Doctorate in Molecular Biology from University of Brasilia in 2009. From 2010 to 2014, she worked in the Bioinformatics Laboratory at the Institute of Biology as a Post-Doctoral Fellow. Since 2014, she has been working as Researcher in the Fiocruz/Manaus.

Marcelo Brigido received his Doctorate in Molecular Biology from University of São Paulo in 1992. Since then, he has been working at the Institute of Biology from the University of Brasilia as an Associated Professor. He has been coordinating the Bioinformatics Laboratory at the Institute of Biology since 2000. Since 2011, he became Full Professor for Molecular Biology at the University of Brasilia.

Maria Emilia M. T. Walter received her Doctorate in Computer Science from University of Campinas in 1999. Since 1988, she has been working as an Adjoint Professor for Computer Science at the University of Brasilia. Since 2000, she has been working in the Bioinformatics Laboratory at the Institute of Biology. She has been participating of the Special Commmittee for Computational Biology of the Brazilian Computer Society since 2005.

Peter F. Stadler received his PhD in Chemistry from U. Vienna in 1990 and then worked as Assistant and Associate Professor for Theoretical Chemistry at the same School. In 2002 he moved to Leipzig as Full Professor for Bioinformatics. Since 1994 he is External Professor at the Santa Fe Institute. He is an External Scientific Member of the Max Planck Society since 2009 and an Corresponding Member Abroad of the Austrian Academy of Sciences since 2010.