INSTITUTO OSWALDO CRUZ

Doutorado em Biologia Computacional e Sistemas

MINERAÇÃO DE TEXTOS CIENTÍFICOS VISANDO À IDENTIFICAÇÃO DE COMPONENTES BIOATIVOS COM POTENCIAL TERAPÊUTICO PARA O TRATAMENTO DE DENGUE, MALÁRIA E DOENÇA DE CHAGAS

MILENE PEREIRA GUIMARÃES DE JEZUZ

Rio de Janeiro

2013

Ministério da Saúde

FIOCRUZ
Fundação Oswaldo Cruz

INSTITUTO OSWALDO CRUZ

Pós-Graduação em Biologia Computacional e Sistemas

MILENE PEREIRA GUIMARÃES DE JEZUZ

Mineração de textos científicos visando à identificação de componentes bioativos com potencial terapêutico para o tratamento de dengue, malária e doença de Chagas

Tese apresentada ao Instituto Oswaldo Cruz como parte dos requisitos para obtenção do título de Doutora em Biologia Computacional e Sistemas

Orientador(es): Dr. Oswaldo Gonçalves Cruz
Dr. Ernesto Raúl Caffarena

RIO DE JANEIRO
2013

J59 Jezuz, Milene Pereira Guimarães de

Mineração de textos científicos visando à identificação de componentes bioativos com potencial terapêutico para o tratamento de dengue, malária e doença de Chagas / Milene Pereira Guimarães de Jezuz. - Rio de Janeiro, 2013.

xv, 160 f. : il. ; 30 cm.

Tese (Doutorado) - Instituto Oswaldo Cruz, Pós-Graduação em Biologia Computacional e Sistemas, 2013.

Bibliografia: f. 76-81

1. Mineração de textos; 2. Doenças negligenciadas. I. Título.

CDD: 616.96016

Ministério da Saúde

FIOCRUZ
Fundação Oswaldo Cruz

INSTITUTO OSWALDO CRUZ

Pós-Graduação em Biologia Computacional e Sistemas

AUTOR: MILENE PEREIRA GUIMARÃES DE JEZUZ

MINERAÇÃO DE TEXTOS CIENTÍFICOS VISANDO À IDENTIFICAÇÃO DE COMPONENTES BIOATIVOS COM POTENCIAL TERAPÊUTICO PARA O TRATAMENTO DE DENGUE, MALÁRIA E DOENÇA DE CHAGAS

ORIENTADOR (ES): Prof. Dr. Oswaldo Gonçalves Cruz
Prof. Dr. Ernesto Raúl Caffarena

Aprovada em: 21/10/2013

EXAMINADORES:

Prof. Dr. Alberto M. R. Dávila - **Presidente**
Prof. Dr. Flávio Codeço Coelho - **Membro**
Prof. Dr. Floriano Paes Silva Junior - **Membro**
Prof. Dr. Antonio Basilio de Miranda - **1º. Suplente**
Prof. Dra. Leticia Miranda Lery Santos - **2º. Suplente**

Aos meus pais Maria Izilda e Alfredo,

ao meu marido Volfran e

aos meus avós (*in memoriam)*.

Agradecimentos

Em primeiro lugar, agradeço a Deus pela inspiração e apoio para prosseguir, mesmo diante do que poderia ter me afastado dos meus objetivos.

Aos meus pais Alfredo e Maria Izilda por todo amor, apoio, carinho e dedicação em todos os momentos de minha vida, algo que posso resumir como amor incondicional e supremo.

Ao meu marido Volfran por ter feito os meus dias mais felizes e ter me suprido das mais diversas formas de toda a força que precisei desde o início do doutorado. Não foram poucos os momentos de desespero em que ele de forma magnífica conseguiu me animar e deixar firme.

Aos meus orientadores Oswaldo e Ernesto: sem o apoio de vocês desde o início, não teria sequer continuado a pós-graduação. Obrigada por acreditarem em mim, pelo respeito, amizade e carinho. E principalmente por entenderem a minha necessidade de continuar no mercado de trabalho paralelamente ao projeto.

Aos amigos que conquistei na Capgemini e que juntos, cada um de sua maneira, me ajudaram a prosseguir desde o início: Leonardo Rezende, meu coordenador, o qual com sabedoria e amizade permitiu minha dedicação ao projeto. Queridos José Roberto, Luiz Albuquerque, Claudio Pacheco, Renata Barboza, Alexandre Martins (e Rafaela), Daniele Sales, Fabiana Teixeira, Allan, Tiago, Thiago e Yann: consegui chegar até aqui ao som das risadas e do carinho fraternal de todos!

Aos que posso chamar de irmãos de coração, pelo carinho e por lembrarem que eu preciso persistir sempre: Elizangela, João Paulo (Lívia e Belinha), Adriana Silva (e Aryane), Débora Corrêa e Gustavo Semaan, Priscila Martins, Simone Mucks (e Sophia), Carina Poswar, Mônica Azevedo e André, Tulla e Paulo Vinicius, Aline e Heitor, Ana Carolina e Flávio, Valéria e Fábio, Claise, Renata Martins e Jefferson, Adriana (e Sophia).

Aos que me ajudaram a caminhar no conhecimento da Biologia com paciência e amizade: Priscila Monnerat e Daniel Loureiro. Adriana Fróes, obrigada por me ajudar nos momentos complicados e não ter me abandonado, auxiliando até nas buscas por orientação de tese. Amanda Sutter, obrigada pelas vezes que me auxiliou no laboratório, nas idas à secretaria, pelo apoio imediato e amigo sempre que precisei.

Aos que durante todos os momentos mais complicados e que levaram a minha ausência, estavam aos pés de Jesus orando por mim como meus anjos particulares: Reverendo Eduardo

Costa e Marcela Guerra, Diácono Luiz Coelho, Gisèle Pimentel, Alcy Zamith (*in memorian*) e todos os irmãos da Paróquia Anglicana da Santíssima Trindade.

Aos grandes companheiros que tive a oportunidade de conhecer ao longo anos de estudos na Fiocruz e no PROCC, que fizeram as aulas mais animadas e agradáveis, além dos trabalhos em grupo mais produtivos: Adriana, Kele, Felipe, Bruno, Artur e Gilberto.

Ao Instituto Oswaldo Cruz, pelo auxílio para que eu viesse a concluir o curso e contribuir com mais um projeto de estudo para a área acadêmica.

A CAPES e PAPES/VI, pelo apoio financeiro que permitiu a conclusão deste trabalho.

À Capgemini, pelo apoio para dar continuidade a esse trabalho.

Aos membros da banca examinadora, por terem aceitado o convite da avaliação desse trabalho e pela certeza da ajuda crítica e construtiva do mesmo.

LISTA DE FIGURAS

LISTA DE TABELAS

Ministério da Saúde

FIOCRUZ
Fundação Oswaldo Cruz

INSTITUTO OSWALDO CRUZ

MINERAÇÃO DE TEXTOS CIENTÍFICOS VISANDO À IDENTIFICAÇÃO DE COMPONENTES BIOATIVOS COM POTENCIAL TERAPÊUTICO PARA O TRATAMENTO DE DENGUE, MALÁRIA E DOENÇA DE CHAGAS

RESUMO

Milene Pereira Guimarães de Jezuz

As doenças negligenciadas, como a dengue, malária e doença de Chagas, entre outras, que prevalecem em países menos desenvolvidos e em ambientes cercados por condições de pobreza, afetam um sexto da população mundial, matando cerca de três mil pessoas a cada dia no mundo. Porém, pouco investimento tem sido feito em pesquisas sobre essas doenças com o fim de obter fármacos menos agressivos aos seres humanos e com ações mais eficazes. Os fármacos existentes utilizados atualmente em tratamentos para essas doenças datam de 30, 40 ou até 50 anos atrás.

Existe um grande volume de trabalhos científicos disponibilizados em bibliotecas digitais que armazenam artigos voltados à descrição da biologia, imunologia e genética dos parasitas que causam estas doenças. Esses trabalhos podem ser acessados através de técnicas para mineração de textos, em busca de compostos bioativos ainda não completamente explorados que venham contribuir para o desenvolvimento de novos tratamentos contra essas doenças.

Com esse fim, neste trabalho é apresentada uma metodologia organizada a partir do *workflow* WIMBAT que utiliza métodos e técnicas de mineração de textos para possibilitar a extração de termos que descrevam tais compostos a partir de informações obtidas em bancos de dados biológicos, culminando com a construção de grafos para possibilitar a análise de associações entre os compostos identificados e a sua função aos agentes causadores destas doenças.

Ministério da Saúde

FIOCRUZ
Fundação Oswaldo Cruz

INSTITUTO OSWALDO CRUZ

TEXT MINING OF SCIENTIFIC TEXTS AIMED AT IDENTIFYING BIOACTIVE COMPONENTS WITH POTENTIAL THERAPEUTIC FOR THE TREATMENT OF DENGUE, MALARIA AND CHAGAS DISEASE

ABSTRACT

Milene Pereira Guimarães de Jezuz

Neglected diseases such as Dengue, Malaria and Chagas disease among others are prevalent in less developed countries and in environments surrounded by poverty, affecting one-sixth of the world population, killing about 3000 people each day worldwide. However, a small investment has been made in research on these diseases to obtain less aggressive drugs to humans and accomplish most effective actions and thus, the existing drugs used in treatments date back 30, 40 or even 50 years back.

There is a large volume of scientific papers available in digital libraries that store articles related to the description of the biology, immunology and genetics of the parasites that cause these diseases and can be accessed through text mining techniques, aiming the search of bioactive compounds not properly exploitedyet that might contribute to the development of new treatments against these diseases.

To this end, this thesis presents the workflow based methodology called WIMBAT that uses methods and text mining techniques to enable the extraction of terms describing such compounds from information obtained from biological databases, ending within the construction of graphs that enable the specialistthe associations analysis between the identified compounds and their function to the causative agents of these diseases.

LISTA DE ABREVIATURAS

AC	Análise de Correspondência
ACM	Association for Computing Machinery
ASCII	American Standard Code for Information Interchange
BioIEDM	Biomedical Information Extraction and Data Mining
BLAST	Basic Local Alignment Search Tool
CID	Classificação Internacional de Doenças
DTD	Data Type Document
EC	Enzyme Commission Numbers
GATE	General Architecture for Text Engineering
IEEExplore	Institute of Electrical and Electronics Engineers
KEGG	Kyoto Encyclopedia of Genes and Genomes
LOINC	Logical Observation Identifiers Names and Codes
MEDLINE	Medical Literature Analysis and Retrieval System Online
MeSH	Medical Subject Headings
NCBI	National Center for Biotechnology Information
NCIUBMB	Nomenclature Committee of the International Union of Biochemistry and Molecular Biology
NLM	U.S. National Library of Medicine
NT1	Nucleoside transporter 1
PDB	Protein Data Bank
PLN	Processamento de Linguagem Natural
PNP	Purine Nucleoside Phosphorylase
RefSeq	Reference Sequence
SGBD	Sistema de Gerenciamento de Banco de Dados
SNOMED	Systematized Nomenclature of Medicine
UMLS	Unified Medical Language System
Uniprot	Universal Protein Resource
UniProtKB	UniProt Knowledgebase
W3C	World Wide Web Consortium
WHO	World Health Organization
WIMBAT	Workflow para Identificação de Moléculas Bioativas em Arquivos de Texto
XSD	XML Schema

ÍNDICE

1. Introdução

1.1 Pesquisas sobre doenças negligenciadas

As doenças negligenciadas, entre elas a dengue, doença de Chagas e malária, prevalecem em países menos desenvolvidos e em ambientes cercados por condições de pobreza e falta de saneamento básico. Elas afetam cerca de um sexto da população mundial, ou seja, mais de um bilhão de pessoas[1], matando cerca de três mil pessoas a cada dia no mundo. Essas doenças contribuem, inclusive, para a manutenção das condições de desigualdade social, uma vez que representam uma força contrária ao desenvolvimento dos países afetados. Além disso, é importante destacar que se trata de um problema de saúde pública.

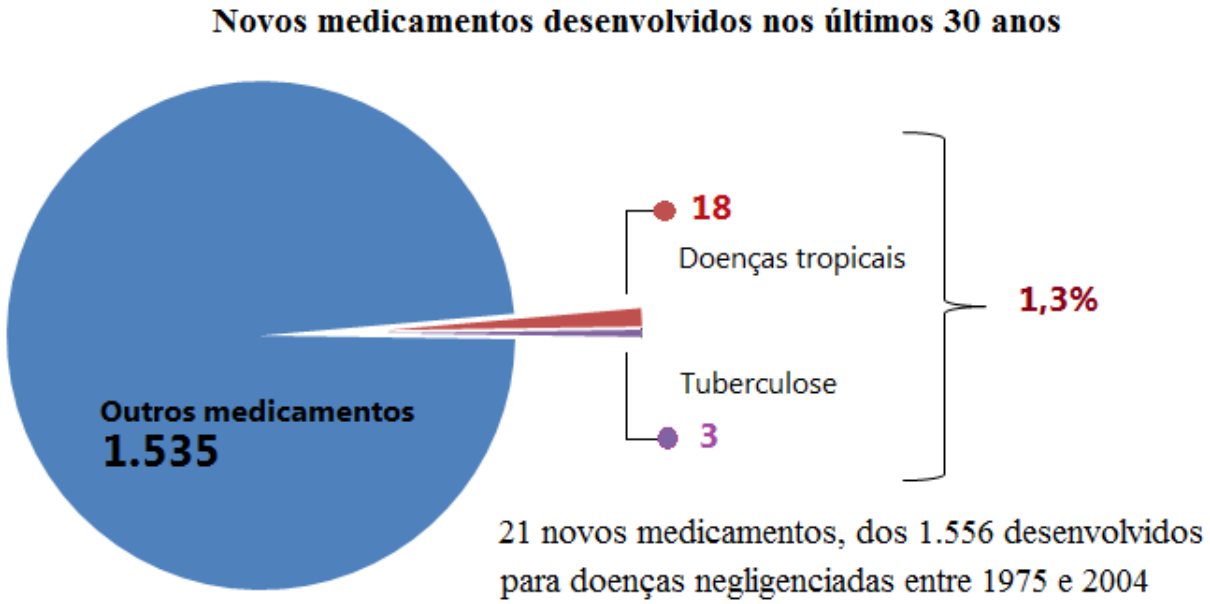

Figura 1.1 Ilustração do registro de novos medicamentos desenvolvidos entre 1975 e 2004. Adaptado de (1)

Pouco se tem investido em pesquisas sobre essas doenças, como é apresentado na Figura 1.1 e, consequentemente, a busca por fármacos menos agressivos aos seres humanos e com ações mais eficazes recebem apenas 5% dos recursos globais para pesquisa e desenvolvimento (P&D), tendo como origem as instituições privadas, como as indústrias farmacêuticas (1).

Uma das parcerias recentemente anunciadas no Brasil para desenvolvimento de medicamentos inovadores, inicialmente contra a Doença de Chagas e a Leishmaniose, envolve a GlaxoSmithKline e Fiocruz. Essa parceria representa um esforço a favor da

[1] Fonte: Dados da Organização Mundial de Saúde (OMS)

produção de novos fármacos contra essas doenças (2), já que os fármacos utilizados em tratamentos datam de 30, 40 ou até 50 anos atrás (e.g. Pentamidina (1952) contra a Leishmaniose tegumentar americana; Nifurtimox (1972) contra a doença de Chagas; artemisinina (1971) contra a malária).

De acordo com o Ministério da Saúde no Brasil, as ações iniciais para P&D relacionadas a doenças negligenciadas foram iniciadas em 2003, através de editais temáticos representando no período de 2003 a 2008 um total de 203 projetos financiados com investimentos da ordem de R$ 10,6 milhões (3).

Atualmente, a iniciativa Medicamentos para Doenças Negligenciadas[2] (DNDi, sigla em inglês) reúne a Fundação Oswaldo Cruz em parceria a institutos de outros países, como o Instituto Pasteur da França, o Ministério da Saúde na Malásia, Institutos de pesquisa médica do Quênia e da Índia e a ONG Médicos sem Fronteiras, de forma a fornecer até 2014 de seis a oito novos tratamentos que atendam às necessidades dos pacientes afetados por Leishmaniose, a doença do Sono, a doença de Chagas e a malária.

Dentre as doenças negligenciadas citadas, o presente trabalho aborda a dengue, doença de Chagas e malária.

1.1.1 Dengue

Dengue é um arbovírus, ou seja, um vírus do gênero Flavivirus, pertencente à família Flaviviridae. De acordo com a Organização Mundial da saúde (OMS) de 40 a 100 milhões de pessoas a cada ano são infectadas por esse vírus (4). Outros membros do mesmo gênero incluem o vírus da encefalite transmitida por carrapatos, vírus da febre hemorrágica de Omsk, o vírus da febre amarela, o vírus do Nilo Ocidental, vírus da encefalite de St. Louis encefalite, vírus da encefalite japonesa e vírus da doença da floresta Kyasanur. A maioria desses vírus é transmitida por artrópodes (mosquitos ou carrapatos)

Até hoje, são conhecidos quatro sorotipos: DENV1, DENV2, DENV3 e DENV4. Quando o homem é infectado, ele desenvolve imunidade permanente ao sorotipo que causou a infecção e imunidade temporária e parcial aos outros sorotipos.

[2]DNDi: http://www.dndi.org.br/

Todos os sorotipos podem levar a quadros graves da doença. Os sintomas incluem dores de cabeça, febre, dores nas juntas e músculos, erupção cutânea característica e em casos mais graves, hemorragias (como o caso da dengue hemorrágica) que podem levar à morte.

O vírus da dengue é transmitido por artrópodes (e.g. mosquitos), especificamente o *Aedes aegypti* (Figura 1.2). Essa espécie é a mais importante na transmissão da doença, que pode ser adquirida através de uma única picada, e também pode ser transmissora da febre amarela urbana (4). Outras espécies de Aedes que transmitem a doença incluem o *A. albopictus*, *A. polynesiensis* e o *A. scutellaris*. Os seres humanos são o principal hospedeiro do vírus.

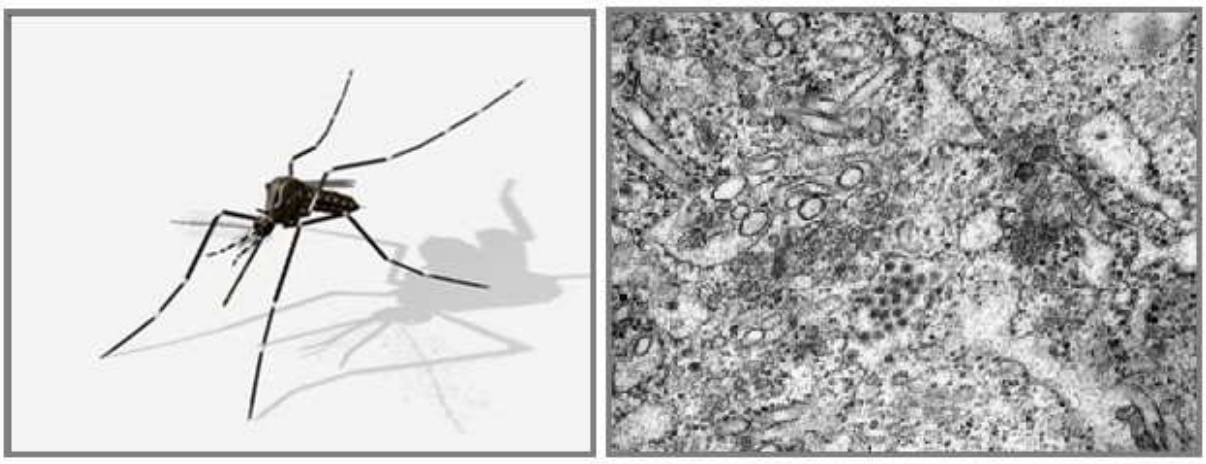

Figura 1.2 Aedes aegypti (modelo virtual 3D produzido pelo setor de Tratamento e Produção de Imagem do IOC[3]) e células do mosquito infectadas pelo vírus da dengue[4] (pontos pretos)

A identificação da infecção através de exames de laboratório é feita através de testes sorológicos, com presença de anticorpos classe IgM (Imunoglobulina M) / IgG (Imunoglobulina G) ou isolando o agente etiológico, que é o método mais específico. Estes dois exames são complementares.

Não há nenhum fármaco antiviral específico para dengue, mas a comunidade científica internacional e brasileira está trabalhando firme neste propósito. A dengue é um desafio para os pesquisadores, pois a sua vacina é mais complexa que as demais por ser necessária a combinação de todos os sorotipos para que se obtenha um imunizante realmente eficaz.

Em setembro de 2012 o laboratório francês Sanofi Pasteur anunciou que os testes da sua nova vacina contra a dengue atingiram a eficácia média de 30%. Contra cada variação do vírus

[3]Fonte: http://www.ioc.fiocruz.br/pages/informerede/corpo/hotsite/dengue/img/mosquisto.jpg
[4]Fonte: http://www.fiocruz.br/ioc/media/28_03_08_virus_dengue_1.jpg

(sorotipo) obteve-se uma taxa de eficiência, ficando entre 60% e 90% para os sorotipos DEN-1, DEN-3 e DEN-4. Para o sorotipo DEN-2 ainda não foi obtida uma vacina eficaz.

O Instituto Butantan anunciou em Dezembro de 2011 a produção de uma vacina com estimativa de que esteja disponível em 2015, seguindo as aprovações das autoridades éticas e regulatórias, conforme anunciado pela Assessoria de Imprensa do Instituto Butantan em (5). A vacina será preventiva e tetravalente protegendo contra os quatro tipos de vírus da dengue e está sendo testada em humanos desde Agosto de 2013, conforme outro pronunciamento encontrado no site do Instituto Butantan (6).

1.1.2 Doença de Chagas

A doença de Chagas (ou tripanossomíase americana) é uma infecção crônica, tornando-se por isso um problema epidemiológico apenas em alguns países da América Latina, embora por conta da migração crescente de populações, tenha aumentado o risco de transmissão por transfusão de sangue até mesmo nos EUA.

É estimado que existam até 18 milhões de pessoas com esta doença, dentre os 100 milhões que constituem a população de risco, distribuída por 18 países americanos. Destes infectados, cerca de 20.000 morrem a cada ano (7).

A doença de Chagas foi nomeada por Carlos Chagas, médico brasileiro quem primeiro descreveu a doença em 1909, assim como o ciclo de vida do parasita (8). As formas habituais de transmissão do protozoário flagelado causador da doença de Chagas, o *Trypanosoma cruzi* (Figura 1.3), são: por transfusão de sangue, via congênita, oralmente através da ingestão de alimentos contaminados e também, mais comumente conhecido, através de um inseto chamado barbeiro (*Triatoma infestans*), que se infecta ao sugar o sangue de um organismo portador da infecção.

O diagnóstico pode ser realizado através de exames laboratoriais com a busca do DNA do parasita pela metodologia PCR (reação em cadeia da polimerase); com a utilização de microscópio para a busca do parasita em amostras do sangue do paciente, o que é possível apenas na fase aguda após cerca de duas semanas após a picada. Esta forma de disgnóstico detecta mais de 60% dos casos nesta fase; ou com a detecção de anticorpos específicos contra o parasita no sangue, sendo os testes sorológicos mais utilizados a imunofluorescência indireta (IFI), hemaglutinação (HAI) e “enzyme-linked immunosorbent assay” (ELISA)(7).

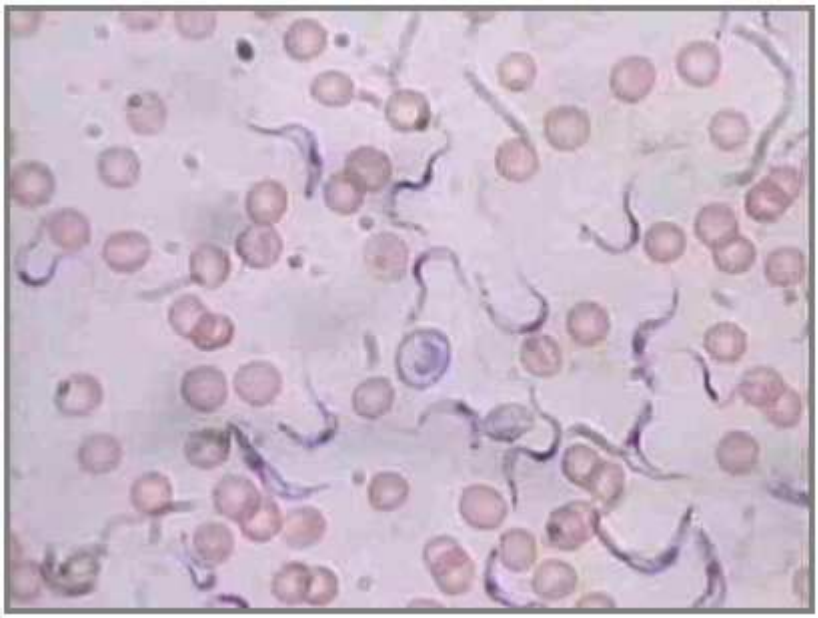

Figura 1.3 Trypanossomacruzi no sangue de um paciente infectado. [5]

Os sintomas da doença de Chagas são febre, aumento do tamanho dos gânglios, aumento do volume do baço e fígado, distúrbios elétricos do coração e/ou inflamação das meninges nos casos mais graves. Na fase aguda, os sintomas duram de três a oito semanas e na crônica, os sintomas estão relacionados a distúrbios no coração e/ou no esôfago e no intestino. Cerca de 70% dos portadores permanecem de duas a três décadas na chamada forma assintomática ou indeterminada da doença.

Para o tratamento desta infecção, na fase inicial aguda, é realizada a administração de fármacos como Nifurtimox, Alopurinol e Benzonidazol, que curam completamente ou diminuem a probabilidade de cronicidade em mais de 80% dos casos. A fase crônica é incurável, já que os danos em órgãos como o coração (Cardiopatia Chagásica Crônica - CCC) e o sistema nervoso são irreversíveis. Nestes casos, apenas tratamentos paliativos podem ser utilizados.

1.1.3 Malária

A malária é considerada uma das infecções parasitárias mais graves da humanidade. Presente em 110 países do mundo, a malária ameaça metade da população mundial. De 350 a 500 mil casos ocorrem em todo o mundo anualmente, principalmente no continente africano, onde mata uma criança a cada 30 segundos segundo estimativa da Organização Mundial da Saúde (OMS) (9). É uma infecção transmitida de pessoa a pessoa através da picada de mosquitos *Anopheles*.

[5] http://www.epub.org.br/svol/imagens/trypanosoma2.jpg

O agente causador é um parasita do filo Apicomplexa, gênero *Plasmodium*. Mesmo já sendo conhecidas mais de 100 espécies deste gênero, apenas o *P. falciparum*, *P. malariae*, *P. ovale* e *P. vivax* (10) infectam humanos (Figura 1.4). Mais de 99% das infecções são causadas por *P. vivax, sendo o P. falciparum,* o mais agressivo (11).

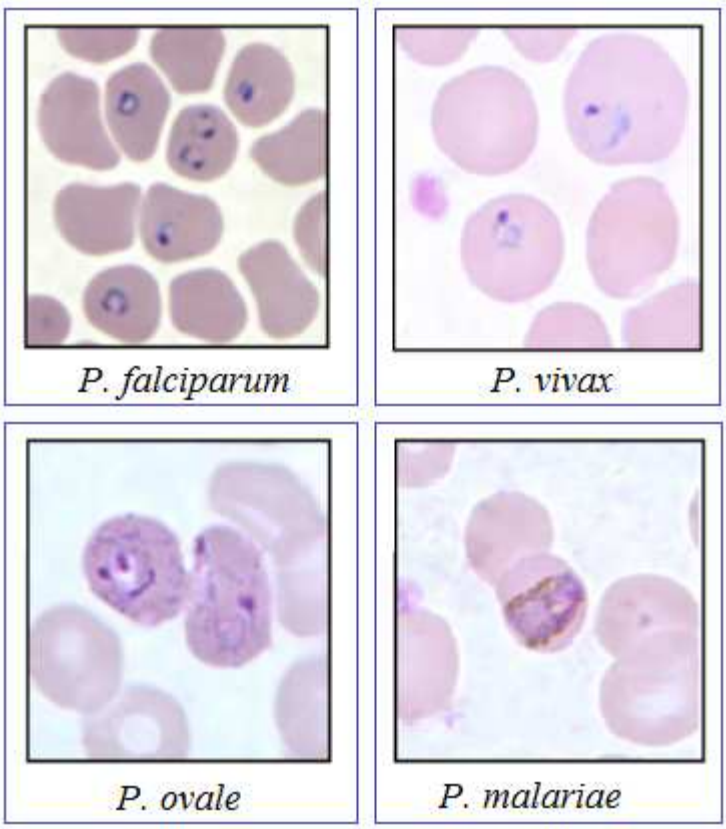

Figura 1.4 Imagens capturadas em esfregaços de sangue contaminado pelosparasitas causadores da malária: Em forma de anel: *P. falciparum* e *P. vivax* na forma de banda; *P. ovale* e *P. malariae* em forma de anel.[6]

Após ter adquirido a infecção, o principal sintoma é a febre constante com periodicidade irregular, mas dores de cabeça, náuseas, hemorragias e fadiga também são sintomas. A doença pode provocar problemas hepáticos, respiratórios, cardiovasculares, cerebrais e gástricos.

Como auxílio ao diagnóstico desta infecção, já que a distribuição geográfica da malária não é homogênea nem mesmo nos países onde a transmissão encontra-se elevada, devem ser resgatadas informações sobre a área de residência ou relato de viagens de exposição ao

[6] Imagens obtidas em http://www.cdc.gov/malaria/images/microscopy/about/falciparum.jpg, http://www.cdc.gov/malaria/images/microscopy/about/vivax.jpg, http://www.cdc.gov/malaria/images/microscopy/about/ovale.jpg e http://www.cdc.gov/malaria/images/microscopy/about/malariae.jpg.

parasita, como em áreas endêmicas (tropicais). De forma complementar, informações sobre transfusão de sangue, compartilhamento de agulhas em usuários de drogas injetáveis, transplante de órgãos podem sugerir a possibilidade de malária induzida.

Paralelamente, os exames laboratoriais que podem ser realizados para diagnóstico da doença são:

a) Gota espessa: baseia-se na visualização do parasita através de microscopia ótica, após coloração com corante vital (azul de metileno e Giemsa), permitindo a diferenciação específica dos parasitos a partir da análise da sua morfologia, e pelos estágios de desenvolvimento do parasito encontrados no sangue periférico;
b) Testes rápidos para detecção de componentes antigênicos de plasmódio: Realizados em fitas de nitrocelulose contendo anticorpo monoclonal contra antígenos específicos do parasito. Apresentam sensibilidade superior a 95% quando comparado à gota espessa, e com parasitemia superior a 100 parasitos/µL;
c) Esfregaço delgado: Método que possui baixa sensibilidade, sendo o único que permite, com facilidade e segurança, a diferenciação específica dos parasitas, a partir da análise da sua morfologia e das alterações provocadas no eritrócito infectado.

O tratamento contra a malária baseia-se na susceptibilidade do parasita aos radicais livres e substâncias oxidantes, morrendo em concentrações destes agentes inferiores às mortais para as células humanas. O quinina (ou o seu isómero quinidina) foi o primeiro antimalárico utilizado e - ainda é utilizada. No entanto, a maioria dos parasitas já é resistente às suas ações. A quinina foi suplantada por fármacos sintéticos mais eficientes, como quinacrina, cloroquina e primaquina.

Em 2010 foi anunciado um fármaco desenvolvido por agências do governo dos Estados Unidos e Cingapura, pesquisadores de universidades da Suíça, da Tailândia, EUA e da Grã-Bretanha, além da companhia farmacêutica Novartis, chamada NITD609 (12). Tal fármaco se mostrou eficaz contra o Plasmodium falciparum e o Plasmodium vivax, apresentando resultados relevantes em animais.

Algumas vacinas estão em desenvolvimento e tem mostrado resultados otimistas. Diversos países e laboratórios se dedicam há mais de uma década a fazer uma vacina com mais de 80% de eficiência usando métodos distintos, sendo que, de acordo com Collins e Barnwell (13), a grande variabilidade e resistência do parasita tem sido um problema difícil de contornar.

2. Justificativa

A dengue, malária e doença de Chagas estão entre as doenças mais letais e por isso são tratadas como doenças crônicas, cuja prevenção e tratamento são custosos. Por prevalecerem em países menos desenvolvidos e em ambientes cercados por condições de pobreza, as pesquisas sobre essas doenças ainda não recebem a atenção que deveria ser oferecida.

Além das prioridades definidas no Brasil para a orientação dos investimentos do Ministério da Saúde já serem um grande passo a favor do avanço das pesquisas, a utilização de repositório de dados digitais para disponibilizar e/ou consultar informações é uma forma de contribuir com o avanço das pesquisas contra essas doenças e oferece uma forma de diminuir custos e o tempo total dos projetos de pesquisa.

Hoje em dia existe um grande volume de trabalhos científicos disponibilizados em bibliotecas digitais como MEDLINE (14), cuja parte do conteúdo de artigos forma o PubMed (15), ACM (16) e IEEExplore (17) que armazenam artigos voltados à descrição da biologia, imunologia e genética dos parasitas que causam estas doenças. Também podem ser encontradas informações registradas no GenBank (18), PDB (19) e KEGG (20), todos eles caracterizados como bancos de dados biológicos que armazenam respectivamente os dados de genes, proteínas e vias metabólicas.

Portanto, com a soma das informações publicadas em artigos que vêm aumentando nos últimos anos conforme mostra a Figura 2.1. As informações em bancos de dados, o volume de informações disponível sobre essas doenças é alto e de crescimento contínuo por conta do avanço das tecnologias disponíveis de alto processamento.

Assim, esse crescimento leva à necessidade de metodologias que permitam a extração do conhecimento de uma maneira ágil e confiável.

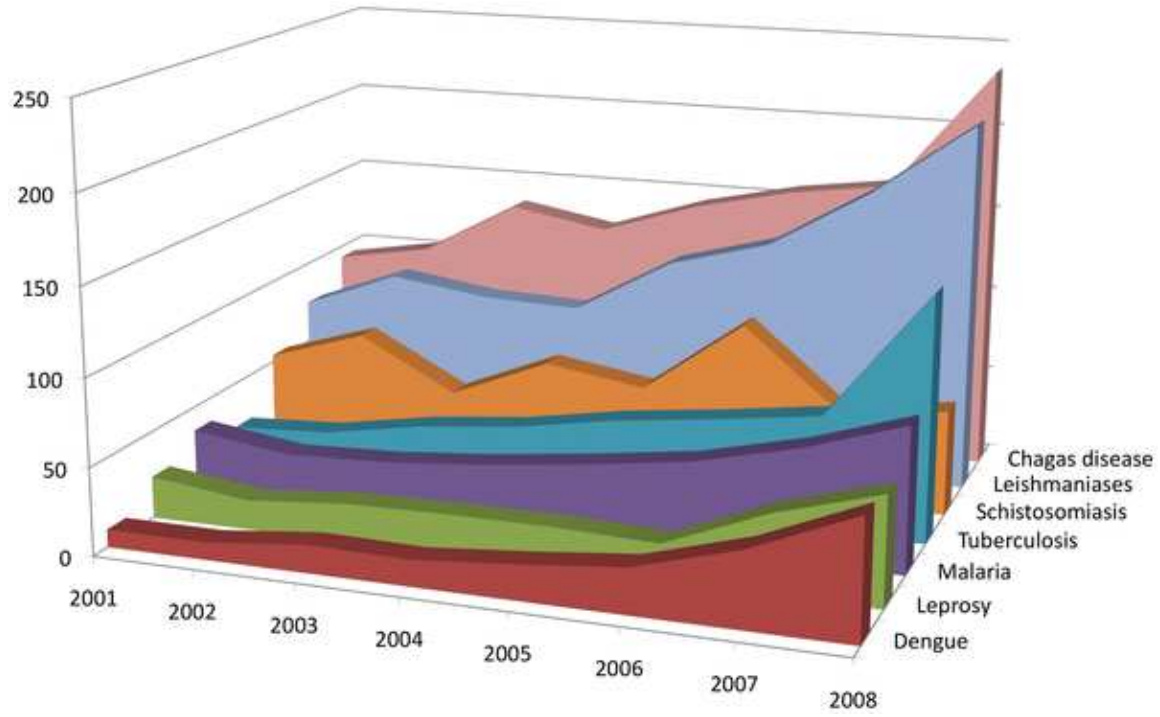

Figura 2.1 Evolução das publicações de autores brasileiros sobre as sete doenças negligenciadas abrangidas pelo DECIT CNPq. Adaptado de Morelet al. (2009) (21)

Em bancos de dados encontramos as informações armazenadas de uma forma estruturada, o que facilita a extração de conhecimentos a partir de consultas específicas disponíveis em Sistemas Gerenciadores de Bancos de Dados (SGBDs) e de algoritmos de *Data Mining* (Mineração de Dados) utilizados para o caso de bancos de dados mais robustos.

Uma das formas utilizadas para registro e recuperação das informações disponibilizadas em bancos de dados é a estrutura XML (*eXtensible Markup Language*) (22). Tal estrutura é uma recomendação da W3C (*World Wide Web Consortium*) (23), que foi criada visando separar o conteúdo da informação da formatação da mesma, apresentar os dados de forma legível e fácil para manipular tanto para humanos quanto para ferramentas lógicas. Assim, consegue-se padronizar e organizar os dados hierarquicamente, permitindo a validação e tipagem dos mesmos a partir de estruturas DTD (*Data TypeDocument*) (24) ou XSD (*XML Schema*) (25), sendo esta última mais utilizada desde 2001.

Já em artigos científicos as informações não são estruturadas e, muitas das ferramentas utilizadas para realizar as buscas dessas informações ainda não conseguem suprir todas as necessidades de pesquisa.

Além dos resultados serem apresentados simplesmente como uma lista que oferece uma visão subjetiva, como descrito no trabalho de Lin Y et al. (26), verificar qual artigo é mais relevante para um caso de estudo se torna um trabalho oneroso. Isso leva muitas vezes, a quem solicitou a busca, verificar apenas os primeiros resultados da lista ou verificar apenas os artigos mais recentes, o que não necessariamente indica que o trabalho é o mais relevante.

Outro dilema consiste em retornar, dentre um conjunto de artigos, a correlação entre o termo utilizado na chave de busca e de outros não indicados. Em outras palavras, caso o solicitante da busca procure pelo nome de uma determinada proteína, a lista de artigos retornada poderá conter o nome desta proteína e também de outras moléculas com as quais essa proteína pode interagir diretamente em um processo de ativação ou inibição da sua função biológica.

Isso nos mostra a necessidade da utilização de tecnologias específicas de forma a tornar a busca mais relevante e específica, evitando sucessivas buscas que podem retornar inúmeras listas de artigos irrelevantes para o caso de estudo.

3. Referencial teórico

Como base para este projeto foram criados os próximos subitens para a apresentação do referencial teórico sobre itens da área de Estudos de Informação, contida na Ciência da Computação: recuperação da informação e mineração de textos, mais especificamente teorias sobre descritores e validadores, o que também levou a necessidade do embasamento biológico sobre proteínas.

3.1 Recuperação da informação

A recuperação da informação (RI) estuda as formas de representação, armazenamento e acesso a itens de informação (27). Porém, tradicionalmente, tal termo é relacionado aos métodos de recuperação de informação contidos em conjuntos de documentos disponíveis, tornando possível a extração da informação de um documento texto não estruturado. E, de acordo com (28), deve responder às necessidades de um determinado conceito, necessitando que os documentos presentes na base de dados sejam submetidos a um tratamento para garantir um rápido acesso à informação.

A extração da informação data historicamente dos anos 70, quando foi criado como um subcampo do campo de recuperação de informação mais geral relacionadas à tarefa de

recuperação *ad hoc*, quando os mesmos documentos/conjunto de textos são recuperados aos solicitantes de uma busca que realizarem as mesmas consultas. Por esta razão, existe uma tendência em relacionar com dualidade a captura/extração e classificação de informação, por partirem da premissa de que os documentos e consultas são intercambiáveis, formando modelos de recuperação de filtragem de dados.

Entretanto, a influência destes modelos de recuperação de filtragem ainda é grande, pois esses modelos de recuperação compartilham muitos problemas comuns, tais como a forma de lidar com palavras e símbolos, como para representar um documento, como representar o retorno da consulta, entender a relevância dos dados e como usar realimentação dos dados relevantes. E para atender a estas dificuldades, surgiram métodos específicos de captura/extração e classificação da informação, conforme descrito por Srivastava e Sahami em (29).

Vários são os métodos de captura/extração de informação já utilizados para identificar em coleções de artigos ou segmentos de texto de artigos (especificamente em resumos, palavras-chave, parágrafos específicos ou mesmo no texto completo) se o artigo pertence a tópicos específicos ou às necessidades expressas na maior parte das buscas solicitadas. Os métodos mais populares, descritos no trabalho de Lin, Y. et al. (26) são o modelo Booleano (*boolean model*) e o modelo vetorial (*vector model*).

O modelo Booleano utiliza operadores de lógica booleana para verificar se existe, em um conjunto de documentos, o termo ou conjunto de termos indicados como entrada de busca, retornando o conjunto de documentos cujo resultado foi verdadeiro para qualquer um dos termos utilizados como entrada. O modelo vetorial utiliza um vetor pré-definido de termos indexados (como palavras-chave) para identificar cada documento. Para cada termo é utilizado um esquema de peso que indica um valor de ocorrência de acordo com sua aparição em cada documento.

Como apoio a essas técnicas, o Processamento de Linguagem Natural (PLN) utiliza métodos para converter as ocorrências de linguagem humana em representações mais formais para posteriormente serem classificadas computacionalmente de acordo com uma área de conhecimento específica, de maneira parecida com o modelo vetorial, sendo uma técnica utilizada em ferramentas como GoPubMed (30) e Textpresso (31).

Outra tecnologia utilizada para a extração das informações é a relacionada à descoberta de conhecimento em bases de dados textuais, que conforme descrito por Ah-Hwee Tan em (32),

geralmente se refere ao processo de extração de padrões interessantes e não triviais ou conhecimento de documentos de texto não estruturados. Ele pode ser visto como uma extensão da mineração de dados ou conhecimento.

E como grande parte do conhecimento acumulado está contida em arquivos texto, a mineração de textos (ou *Text Mining*) (33) tornou-se uma atividade muito utilizada para a recuperação de conhecimento em várias áreas.

3.2 Text mining

A mineração de texto está relacionada à procura de padrões no texto: É o processo de análise de texto utilizada para extrair informações, que é útil para fins específicos, a partir de conteúdos desestruturados, amorfos e com dificuldade para extrair informações, pois não estão redigidas de uma maneira que é passível de processamento automático. Com isso, a mineração de texto constitui-se de atividades que permite transformar essas informações de uma forma adequada para o consumo por computadores ou por pessoas com necessidade de extrair o significado do texto de uma forma mais ágil, conforme Witten e Frank em (34).

De acordo com Witten I.H. et al. (33), *Text Mining* é uma técnica que tenta recolher informações significativas do texto em linguagem natural. Ela pode ser caracterizada como o processo de análise de textos para extrair informação útil para propósitos específicos ou mais generalizados, transformando textos em índices significativos que podem ser incorporados em outras análises, como projetos de mineração de dados preditivos, aplicação de métodos de aprendizagem não supervisionada (agrupamento), entre outros.

O *Text Mining* engloba um conjunto de fases para recuperação da informação, que de acordo com Michael Goebel e Le Gruenwald em (35), tais fases são:

a) Entendimento do domínio de aplicação e definição do objetivo do processo de descoberta;
b) Aquisição ou seleção do conjunto de dados;
c) Integração e verificação do conjunto;
d) Limpeza dos dados (pré-processamento e transformação);
e) Desenvolvimento de um modelo inicial ou construção de hipóteses;
f) Escolha e aplicação de métodos de mineração;
g) Visualização e interpretação dos resultados;
h) Teste e validação das hipóteses (pode-se refazer parte do processo);

i) Uso e manutenção do conhecimento descoberto (tomada de decisão no domínio).

De forma a auxiliar a fase de “Limpeza dos dados” tem-se conceitos chamados de *Stopwords*, *Keywords*, *Collocations*, *Stemming* e *Corpus*. Eles são utilizados na extração da informação de forma a tornar um texto não estruturado em uma informação semiestruturada, com a identificação de algum padrão, permitindo a utilização de outras ferramentas computacionais para a extração de informações. Tais conceitos são definidos como:

a) *Stopwords* compreende uma lista de palavras do mesmo idioma dos textos da coleção, com uso frequente no idioma, mas cujo significado pouco ou nada contribui para agregar significado ao conteúdo do texto, pois estão presentes em abundância nos textos. Remover estas palavras na etapa de indexação é uma boa prática, por evitar despedício de espaço na construção de índices. A lista das *stopwords* é conhecida também como *stoplist* ou dicionário negativo.
b) *Keywords* são tidas como as palavras de maior importância no texto e estas sejam detectadas com melhor precisão, faz-se necessário a remoção das *stopwords*. Essa importância pode ser verificada a partir do cálculo da freqüência de ocorrências da palavra no texto, o que caracteriza a freqüência absoluta. Essa importância pode ser mais apurada quando feito o cálculo da freqüência relativa, que baseia-se na razão da freqüência absoluta em relação ao número de palavras no texto.
c) *Collocations* são expressões compostas, cujo valor está relacionado aos significados das palavras do agrupamento e de uma semântica extra. Sendo tido também como uma forma de mapeamento do significado das palavras na composição em um significado determinado por alguma área de conhecimento.
d) *Stemming* é um processo de tratamento de palavras para que as mesmas, ao final do processo, estejam de acordo com um padrão chamado *stem*. Este padrão diz respeito à análise da palavra após a remoção de afixos, ou seja, a análise do radical da palavra e assim, chegue-se a um padrão que capture a palavra com o máximo de precisão.
e) *Corpus* é um conjunto de dados lingüísticos (pertencentes ao uso oral ou escrito da língua, ou a ambos), sistematizados segundo determinados critérios, suficientemente extensos emamplitude e profundidade, de maneira que sejam representativos da totalidade do uso lingüístico ou de algum de seus âmbitos, dispostos de tal modo que possam ser processados por computador, com a finalidade de propiciar resultados vários e úteis para a descrição e análise (36).

Tais conceitos fazem parte do processo de descoberta do conhecimento e podem ser aplicados com o auxílio de expressões regulares: Uma expressão regular é uma forma de notação que deriva do trabalho de Stephen Kleene (37), que a utilizou como uma forma de representação de expressões sobre automatos finitos. Também chamada de regex (abreviação do inglês regular expression), permite identificar cadeias de caracteres de uma forma flexível e concisa. Essas cadeias de caracteres podem ser um conjunto de caracteres particulares, palavras ou padrões de caracteres.

Uma expressão regular é escrita em uma linguagem formal, comumente interpretada por um processador de expressão regular, que pode atuar como um analisador sintático ou como um identificador de partes de um texto que correspondem com a especificação dada.

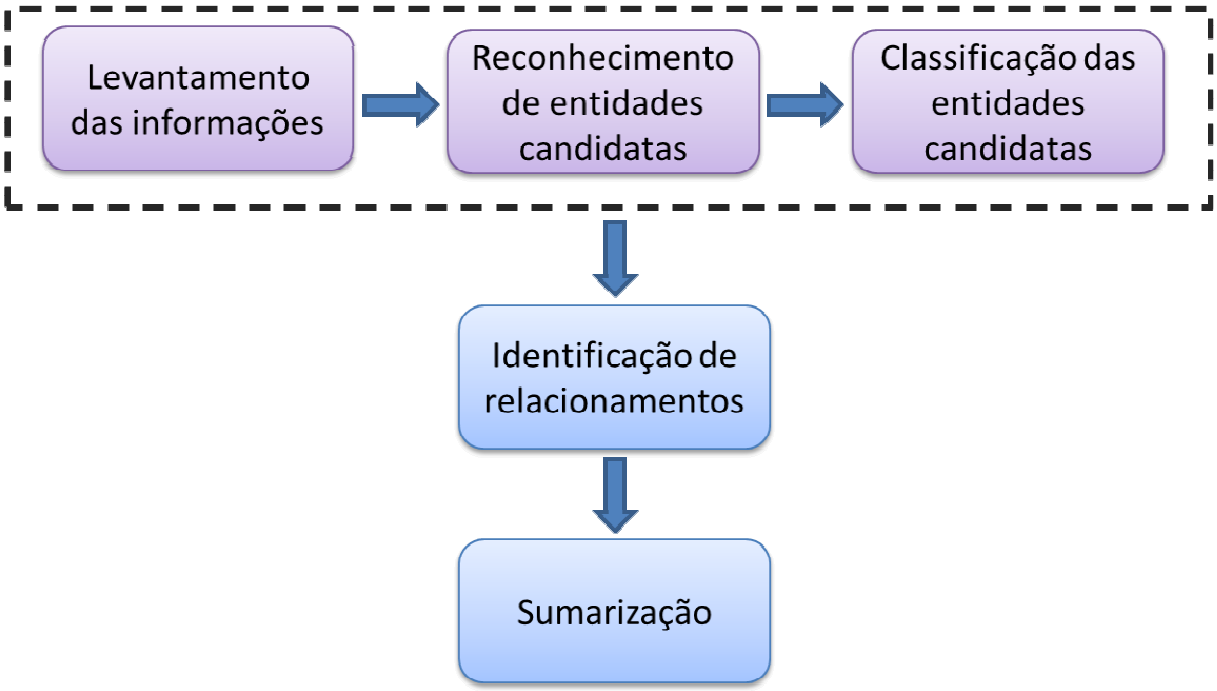

Figura 3.1 Atividades do processo do *Text Mining*

Como resultado da execução das atividades, o *Text Mining* permite que a busca por informações descritas em textos (não estruturados ou semi-estruturados) possam ser acessadas automaticamente. Essa busca antes era feita manualmente, o que se tornava uma tarefa cansativa, demorada e que muitas vezes podia acarretar ao desconhecimento de informações provavelmente pertinentes para a conclusão de um projeto de pesquisa, além de permitir a descoberta de conceitos-chave e grupos similares de documentos, sem que haja necessidade prévia de leitura integral dos documentos.

Atualmente, existem muitas ferramentas computacionais que utilizam os métodos do *Text Mining*, como o **VantagePoint** (38) que minera os textos fornecidos como entrada após serem importados de bibliotecas digitais, e o Bio-IEDM (39) e GATE (40) que buscam termos e associações entre palavras específicas em bibliotecas digitais.

Para validar os termos extraídos dos textos e as associações deles com os métodos de *Text Mining* são utilizados bancos de dados biológicos ou listas de termos/descritores relacionados a esses bancos de dados.

3.3 Validadores de dados biológicos

3.3.1 Bancos de dados biológicos

Os bancos de dados biológicos permitem organizar os dados obtidos em experimentações biológicas de forma que a informação seja disponibilizada da maneira mais simples e podem ser classificados de acordo com os dados que armazenam/disponibilizam (primários, secundários e compostos) ou de acordo com o método de validação dos dados (curados manualmente ou automaticamente).

Bancos de dados primários contêm informações que foram obtidas a partir do sequenciamento de proteínas ou aminoácidos. As informações neles contida podem possuir redundâncias, tendo em vista possuirem alguma interpretação dos dados, mas não uma análise cuidadosa dos mesmos. O UniProtKB (41) (sequencias de proteínas), GenBank (18) (sequências de genomas) e PDB (19) (estrutura de proteínas) são exemplos de banco de dados primários.

Bancos de dados secundários contêm informações derivadas ou que complementam as informações de bancos de dados primários, como sequencias conservadas, assinaturas da sequência, dentre outros, que são originados a partir do alinhamento entre sequencias. E nesses bancos as anotações presentes já não possuem mais redundancias, por terem sido validadas (curadas) manualmente por um especialista. Como exemplo dessa classificação temos o RefSeq (42) e KEGG (20).

Por fim, bancos de dados compostos como o próprio nome já indica, reúnem a combinação de várias informações originadas a partir de bancos de dados primários, porém incluindo consultas a vários outros repositórios incorporando informações de outros bancos de dados primários. O NCBI (*National Center for Biotechnology Information*) (43) pode ser classificado como um banco de dados composto, oferecendo o acesso sobre várias informações tendo como chave de busca (ou entrada) os dados de nucleotídeos e proteínas.

O termo bancos de dados curados, de acordo com Buneman et al. (44), é uma classificação normalmente reservada para aqueles bancos cujo conteúdo (muitas vezes sobre um assunto especializado) foi coletado por exaustivo esforço humano através da consulta, verificação e agregação de fontes existentes, e da interpretação de novos dados brutos (muitas vezes obtidos experimentalmente). Quando curados manualmente, precisa-se da intervenção humana para a validação dos dados. Quando envolve apenas a utilização de processos automatizados, tais bancos são chamados de curados automaticamente.

3.3.2 Descritores

Os descritores permitem verificar se uma palavra encontrada no texto faz parte de um conjunto de termos específicos sobre uma entidade biológica, como também permitem verificar a presença de associações antes não exploradas entre essas entidades.

O CID (Classificação Internacional de Doenças) (45), UMLS (*Unified Medical Language System*) (46), LOINC (*Logical Observation Identifiers Names and Codes*) (47), SNOMED (*Systematized Nomenclature of Medicine*) (48) são exemplos de descritores encontrados na literatura, sendo o MeSH (*Medical Subject Headings*) (49) um dos mais utilizados.

O MeSH é um vocabulário controlado do NLM (*U.S. National Library of Medicine*)(50) organizado em uma estrutura alfabética e hierárquica de termos, utilizado para indexar as publicações armazenadas no PubMed (15), uma biblioteca digital composta por mais de 20 milhões de citações de literatura biomédica, a partir da padronização dos termos associados às palavras-chave. Utilizando esse descritor é possível procurar por artigos científicos no PubMed (15), o que torna a consulta mais específica.

Pesquisas realizadas no MEDLINE (14), sem a utilização de tais termos, tornam a busca mais ampla e genérica, por retornar uma quantidade maior de artigos, sendo necessário maior tempo computacional e/ou manual para a curagem dos termos encontrados.

As descrições utilizadas em vocabulários criados com os recursos da ontologia também são utilizadas como validadores. Ontologia é definida, de acordo com Gruber em (51) como “uma especificação formal de uma conceitualização”.

Uma ontologia consiste do conjunto de instâncias, classes, conceitos e relacionamentos, que descrevem os conceitos aplicados a alguma área. Na área de ciência da computação as ontologias são aplicadas em diversas áreas de conhecimento como a web-semântica,

engenharia de requisitos, inteligência artificial, banco de dados, Biomedicina, Bioinformática, etc. No caso do GoPubMed (30), já citado anteriormente, a Ontologia aplicada para a classificação das informações e auxiliar nas buscas é o Gene Ontology (GO), que é a Ontologia mais disseminada na área de Bioinformática (52).

3.3.3 Validadores de termos biológicos

Na literatura encontra-se trabalhos que utilizam o *Text Mining* como um método de verificação de relacionamentos entre entidades biológicas e de informações que complementam estudos anteriores sobre a dengue e outras doenças.

Jesmin (53) tem proposto o **Epiphany**, uma arquitetura que utiliza o *Text Mining* para gerar uma rede de anotação de interações biológicas através de dados extraídos de resumos de artigos disponibilizados no PubMed.Como resultado, a arquitetura realizou a predição do mecanismo do vírus da dengue apenas com o auxílio do *Text Mining* na literatura selecionada.

No trabalho de Al-Mubaid e Singh (54) foi proposta a utilização do *Text Mining* como um método para a descoberta de associações entre proteínas e doenças de acordo com as informações extraídas de resumos de artigos disponibilizados no MEDLINE (14).

LAITOR (55) foi um projeto desenvolvido como uma estratégia de mineração de texto que permite que conceitos biológicos possam ser pesquisados de acordo com a coocorrência das entidades biológicas (e.g genes e proteínas) ao longo do texto. Suas informações estatísticas são utilizadas amplamente por outra ferramenta chamada **PESCADOR** (56).

A ferramenta **PESCADOR** realiza a busca de biointerações entre as entidades biológicas (e.g interações proteína-proteína), utilizando dicionários pré-compilados de termos depositados em bancos de dados biológicos e dicionários de conceitos biológicos, tendo como entrada uma lista dos artigos de interesse para iniciar a análise.

Nos trabalhos descritos acima (53, 54) a utilização do *Text Mining* se mostrou como uma abordagem promissora para o avanço dos processos de busca para a construção de modelos computacionais que possam descrever (*in silico)* as funções das entidades biológicas causadoras de uma doença e os relacionamentos entre essas entidades.

Para a construção desses modelos é necessário encontrar uma correlação entre os termos, de forma a classificá-los de acordo com um padrão já conhecido. Através de técnicas estatísticas de análise exploratória de dados, essa correlação entre os termos pode ser feita a partir da

Análise de Correspondência (57). Trata-se de uma técnica utilizada para analisar adequadamente tabelas de duas ou múltiplas entradas, levando em conta algumas medidas de correspondência entre linhas e colunas.

Basicamente, este método permite estudar as relações e semelhanças existentes entre as categorias de linhas e entre as categorias de colunas de uma tabela de contingência. Este método pode mostrar, por exemplo, em uma tabela cujas linhas e colunas identificam termos extraídos de textos, a forma em que as variáveis estão relacionadas de acordo com alguma característica (matriz de correspondência) e não somente relacionadas pela sua a existência na tabela, onde a matriz de correspondência é obtida dividindo-se todas as células da tabela de contingência pelo total geral n=312 (Tabela 3.2), formando a proporção da combinação das categorias de variáveis e do total das categorias em relação ao total de unidades que foram classificadas.

Tabela 3.1 Exemplo de tabela de contingência entre artigos e categoria de termos

	Categorias de Termos			
Artigos	**T1**	**T2**	**T3**	**Totais**
A1	5	7	2	14
A2	18	46	20	84
A3	19	29	39	87
A4	12	40	49	171
A5	3	7	16	26
Totais	57	129	126	312

Onde

T1: Termos relacionados a proteínas
T2: Termos relacionados a ligantes
T3: Termos relacionados a fármacos

Tabela 3.2 Exemplo de matriz de correspondência

	Categorias de Termos			
Artigos	T1	T2	T3	**Totais**
A1	0,016	0,022	0,006	0,044
A2	0,058	0,147	0,064	0,269
A3	0,061	0,093	0,125	0,279
A4	0,038	0,128	0,157	0,548
A5	0,010	0,022	0,051	0,083
Totais	0,183	0,412	0,403	1

Esse tipo de análise é interessante no caso de dados que podem ser linearmente separáveis, ou seja, sendo dispostos em um hiperplano, podem ser separados através de uma reta (58).

Outra metodologia de classificação é a de clusterização (59) que basicamente permite realizar o agrupamento ou segmentação de um conjunto de dados, gerando grupos de objetos (*clusters*) de acordo com a similaridade entre eles de acordo com algum critério pré-definido, que os identifica como similares para pertencerem a um grupo, porém dissimilares a ponto de não pertencerem a um mesmo grupo.

Na Figura 3.2 é demonstrado o resultado da execução de um algoritmo de clusterização chamada mean-shift realizada no trabalho de D. Comaniciu. e P. Meer (60), onde são apresentados um conjunto de 110.400 dados clusterizados em sete grupos distintos.

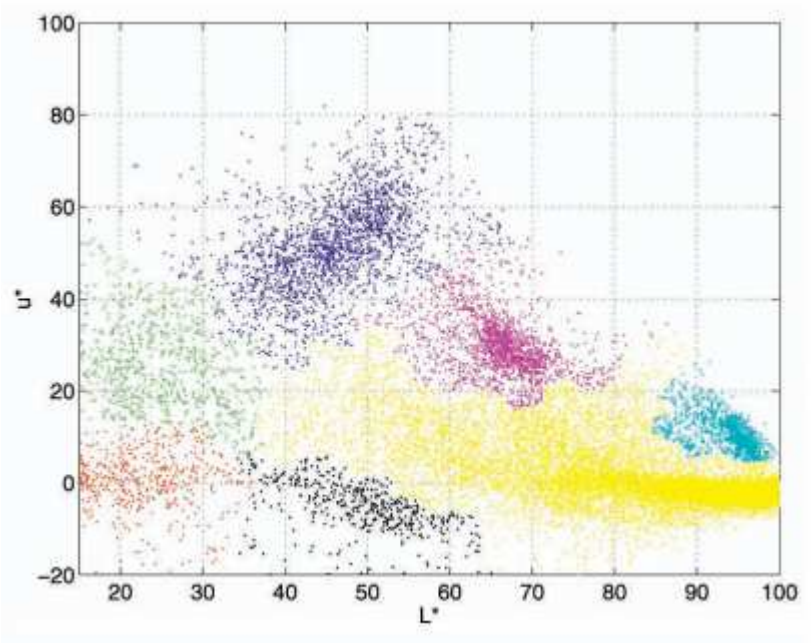

Figura 3.2 Conjunto de 110.400 dados clusterizados em sete grupos distintos(60)

Para complementar os resultados obtidos, é possível a utilização de metodologias que permitem a visualização e a análise destes dados. Para tal finalidade é possível a construção de grafos, como uma forma de apoio à curagem das informações encontradas sobre as possíveis associações entre termos, como por exemplo, as moléculas.

Grafo é uma representação gráfica das relações existentes entre elementos de dados, sendo definido em um espaço de N dimensões como um conjunto V de vértices e um conjunto A de curvas contínuas (arestas). Um exemplo de um gráfico se mostra na Figura 3.3.

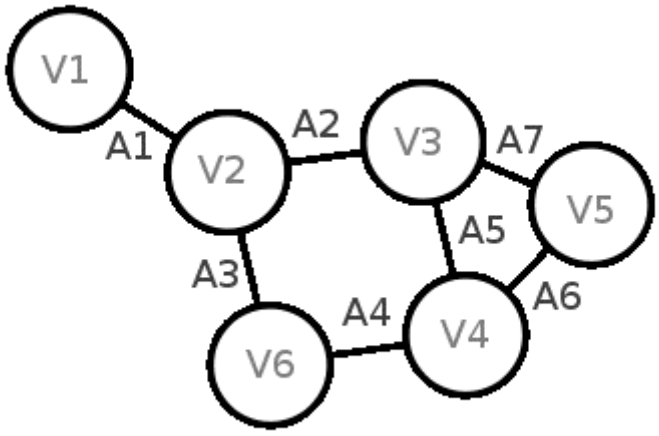

Figura 3.3 Grafo definido em 2 dimensões com 6 vértices (V) e 7 arestas (A)

Com essa metodologia é possível a construção de redes (e.g redes genômicas, redes de proteínas, redes químicas) onde os vértices representam as entidades biológicas e as arestas representam o tipo de associação entre essas entidades.

Um exemplo de utilização dessa metodologia pode ser encontrado no trabalho de Adrien Coulet et al. (61), onde foi proposta a utilização de grafos para a construção de redes semânticas entre termos (entidades biológicas) e o relacionamento entre eles, extraídos de resumos de artigos no MEDLINE para posterior curagem. Com as informações descritas em tal rede é possível visualizar a interação entre genes e fármacos, pois é gerada uma rede de Farmacogenômica, uma área que estuda a relação entre a variação genética e a variação na resposta a fármacos, de forma a identificar importantes associações entre eles e entidades moleculares (62).

As abordagens acima descritas mostraram que a utilização dos métodos de *Text Mining*, classificadores (Análise de Correspondência e Clusterização) e teoria de grafos permite o

acesso a informações do conteúdo de artigos, normalmente realizada manualmente, através de buscas em repositórios de artigos científicos analisados de forma semi-automática.

Porém, as abordagens são genéricas, ou seja, não estão direcionadas a um propósito específico como a obtenção de informações dirigidas apenas a uma doença e/ou entidade biológica. Obviamente, a criação de propostas genéricas pode auxiliar a mais de um projeto de estudo, porém a especificidade nas buscas de acordo com uma área de conhecimento pode aumentar o grau de sucesso do processamento dos dados já que o escopo é diminuído.

Uma abordagem de busca desse tipo envolve a análise das informações sobre proteínas que fazem parte do grupo das enzimas, pois podem ser utilizadas como alvo para novas terapias contra doenças de interesse.

3.4 Proteínas

Proteínas são macromoléculas biológicas cuja importância está relacionada com as funções que desempenham. A composição de uma proteína é complexa e nela encontramos centenas ou milhares de moléculas unidades constituintesmais simples chamadas de aminoácidos, cujo alfabeto é composto por 20 símbolos como descrito na Tabela 3.3.

Tabela 3.3 Lista de aminoácidos que compõem uma proteína

Nome	Nomenclatura	Símbolo	Abreviação
Alanina	Ácido 2-aminopropiônico ou Ácido 2-amino-propanóico	Ala	A
Arginina	Ácido 2-amino-4-guanidina-n-valérico	Arg	R
Asparagina	Ácido 2-aminossuccionâmico	Asn	N
Aspartato ou Ácido aspártico	Ácido 2-aminossuccínico ou Ácido 2-amino-butanodióico	Asp	D
Cisteina	Ácido 2-bis-(2-amino-propiônico)-3-dissulfeto ou Ácido 3-tiol-2-amino-propanóico	Cys, Cis	C
Fenilalanina	Ácido 2-amino-3-fenil-propiônico ou Ácido 2-amino-3-fenil-propanóico	Phe ou Fen	F
Glicina ou Glicocola	Ácido 2-aminoacético ou Ácido 2-amino-etanóico	Gly, Gli	G
Glutamato ou Ácido glutâmico	Ácido 2-aminoglutárico	Glu	E
Glutamina	Ácido 2-aminoglutarâmico	Gln	Q
Histidina	Ácido 2-amino-3-imidazolpropiônico	His	H
Isoleucina	Ácido 2-amino-3-metil-n-valérico ou ácido 2-amino-3-metil-pentanóico	Ile	I
Leucina	Ácido 2-aminoisocapróico ou Ácido 2-amino-4-metil-pentanóico	Leu	L
Lisina	Ácido 2,6-diaminocapróico ou Ácido 2, 6-diaminoexanóico	Lys, Lis	K
Metionina	Ácido 2-amino-3-metiltio-n-butírico	Met	M
Prolina	Ácido pirrolidino-2-carboxílico	Pro	P
Serina	Ácido 2-amino-3-hidroxi-propiônico ou Ácido 2-amino-3-hidroxi-propanóico	Ser	S
Tirosina	Ácido 2-amino-3-(p-hidroxifenil) propiônico ou paraidroxifenilalanina	Tyr, Tir	Y
Treonina	Ácido 2-amino-3-hidroxi-n-butírico	Thr, The	T
Triptofano	Ácido 2-amino-3-indolpropiônico	Trp, Tri	W
Valina	Ácido 2-aminovalérico ou Ácido 2-amino-3-metil-butanóico	Val	V

As proteínas podem ser classificadas de acordo com características específicas como o relacionamento evolutivo, similaridade existente entre elas e/ou a função que desempenham. Esta última característica justifica a escolha das proteínascomo alvos de fármacos quando desempenham a função de enzimas (melhor descritas no item 3.4.1).

De acordo com o trabalho de Singh S. et al. (63), as enzimas representam quase 50% do conjunto de potenciais alvos de fármacos específicos conforme mostra a Figura 3.4, onde Singh et al. também incluem outras opções para alvos de fármacos como GPCRs (Receptores de proteínas – G-proteina – acoplados a proteínas, do inglês “Receptor proteins - G-protein–coupled receptors”), o RNA alvo, enzimas do metabolismo intermediário, sistemas para a replicação do DNA, aparelho de tradução ou proteínas de reparo e membrana.

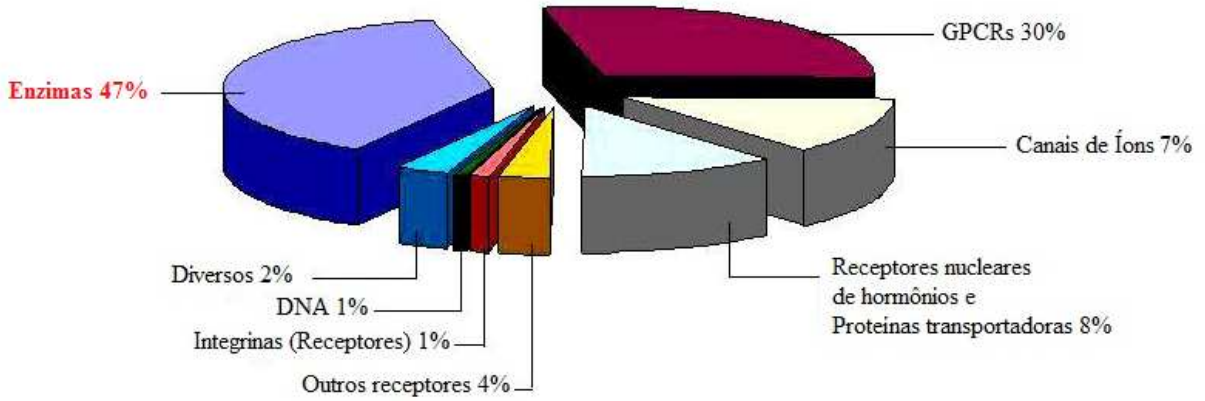

Figura 3.4 Proporção de escolha de potenciais alvos de fármacos: cerca de 50% dos alvos selecionados são as enzimas, conforme estudo de Singh S. et al. (63).

3.4.1 Função das proteínas

As proteínas são classificadas de acordo com as funções que elas desempenham em uma célula, funções essas relacionadas à arquitetura celular, transporte, regulação, entre outras (64). Dentre as várias classificações, as que serão abordadas nessa seção são: Proteínas Estruturais, Transportadoras, Proteínas Reguladoras e Enzimas.

i. Proteínas Estruturais: são aquelas que fornecem firmeza e proteção aos organismos (e.g colágeno, encontrado em cartilagens e tendões).
ii. Proteínas Transportadoras: Podem ser encontradas nas membranas plasmáticas e intracelulares de todos os organismos e sua função é transportar íons, moléculas e macromoléculas através das membranas celulares.
iii. Proteínas Reguladoras: são aquelas envolvidas nos processos de regulação de expressão gênica, auxiliando na regulação de inúmeras atividades metabólicas, como por exemplo, os hormônios insulina e o glucagon, que possuem função antagônica no metabolismo da glicose.
iv. Enzimas: As enzimas fazem parte do grupo mais abundantede proteínas e são responsáveis por catalisar com alta especificidade praticamente todas as reações de um organismo, estando assim frequentemente relacionadas com o controle do metabolismo. A maioria das reações bioquímicas em que as enzimas estão relacionadas envolve a transformação de substâncias, onde aquelas que serão transformadas são chamadas de substrato e ao resultado dessa transformação se denomina substâncias produto.

A quantidade de enzimas conhecidas (milhares) levou à necessidade de criação de classes de enzimas de acordo com o tipo de reação química que catalisam. São seis classes, cujos nomes

foram normatizados pelo comitê NC-IUBMB (*Nomenclature Committee of the International Union of Biochemistry and Molecular Biology*) (65). Tais classes e subclasses de proteínas podem ser catalogadas através do número EC (do inglês *Enzyme Commission Numbers).* Trata-se da numeração mais comumente utilizada para essa forma de classificação. Na Tabela 3.4 estão listadas as seis classes e respectivas reações químicas que catalisam.

Tabela 3.4 Classes de enzimas de acordo com o tipo de reação química que catalisam

Classe 1 (EC 1): Oxidorredutases	Enzimas que catalisam reações de oxidorredução.
Classe 2 (EC 2): Transferases	Responsáveis pela transferência de grupos químicos entre moléculas.
Classe 3 (EC 3): Hidrolases	Utilizam a água como receptor de grupos funcionais vindas de outras moléculas.
Classe 4 (EC 4): Liases	Realizam a formação ou destruição de várias ligações químicas, respectivamente retirando ou adicionando grupos funcionais.
Classe 5 (EC 5): Isomerases	Enzimas que facilitam a transferência de grupos funcionais intramoleculares específicos.
Classe 6 (EC 6): Ligases	Catalisam a formação de uma ligação entre duas moléculas de substrato.

Assim, cada classe divide-se, por sua vez, em subclasses, também numeradas. As subclasses definem em que tipo de grupo as enzimas podem atuar. Por exemplo, um determinado grupo de enzimas pode pertencer à subclasse EC 2.1, que engloba "enzimas que transferem grupos contendo um átomo de carbono".

3.4.2 Utilização de enzimas como alvos terapêuticos

As enzimas são utilizadas como alvos terapêuticos de acordo com a sua função. Uma enzima inibidora desempenha a função de diminuir a atividade de outra enzima ao se acoplar a ela, interagindo com o chamado sítio regulatório, provocando uma mudança na sua conformação e a posterior desativação do sítio catalítico. Já uma enzima ativadora provoca o efeito oposto.

Para algumas enzimas não existe a necessidade de componentes adicionais para que a sua atividade enzimática seja completa, porém algumas outras precisam se ligar a moléculas não protéicas para exercerem a sua atividade. Essas outras moléculas são denominadas de cofatores os quais podem ser inorgânicos ou orgânicos. Os cofatores inorgânicos são os íons metálicos (e.g Zn^{+2}, Mg^{+2}, Mn^{+2}, Fe^{+2}, Cu^{+2}, K^{+1}e Na^{+1}) que interagem através de ligações não covalentes coordenadas.

Os cofatores orgânicos são também chamados de coenzimas. As coenzimas são moléculas relativamente pequenas ao serem comparadas com o tamanho da enzima e fazem parte do sítio ativo da enzima. Portanto, sem elas, a enzima não consegue desempenhar a sua função. Quando as enzimas atuam em conjunto em um organismo, elas fazem parte de vias metabólicas, de forma a acelerarem o processo metabólico das quais fazem parte.

3.4.3 Vias metabólicas

Entende-se por vias metabólicas o conjunto formado por sequências de reações químicas interconectadas responsáveis pelo metabolismo de um organismo, com a finalidade de prover à célula o produto que ela necessita em determinado momento, utilizando energia apenas quando necessário (64).

Uma via metabólica envolve o conjunto de substratos e enzimas. Neste caso, os substratos são moléculas nas quais as enzimas agem para acelerar reações químicas envolvendo os mesmos. Essas reações estão relacionadas à produção ou consumo de energia em determinado momento da via metabólica, para culminar em um novo produto que a célula necessita.

Através da ação de enzimas regulatórias, as vias metabólicas são altamente coordenadas para prover um harmonioso efeito recíproco dentro das várias atividades necessárias para sustentar a vida da célula (64).

Um exemplo de via metabólica é mostrado na Figura 3.5. Trata-se da via metabólica da tiamina, que está presente como uma via do metabolismo da purina sendo parte do estudo de um experimento cerebral em camundongos, onde a malária cerebralé suprimida pela interrupção do transportador NT1 (*Nucleosidetransporter 1*) e não da fosforilase de nucleotídeo da purina PNP (*purinenucleosidephosphorylase*), vias que desempenham principais papéis no salvamento de purinas no parasita causador da malária (*Plasmodiumfalciparum*), como descrito em (66).

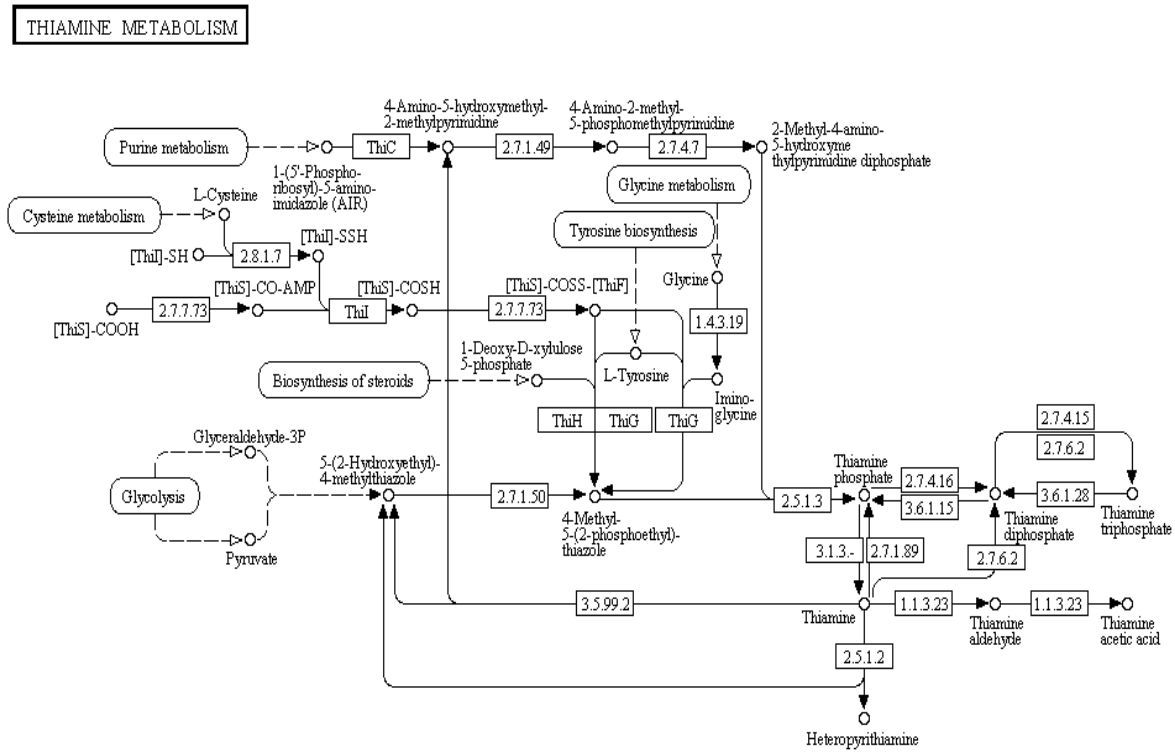

Figura 3.5 Via metabólica da Tiamina[7]

Muita importância é dada às informações coletadas sobre as enzimas e sobre como desempenham suas funções em vias metabólicas e, portanto, elas são fonte para numerosos estudos, sendo um deles a mineração de alvos bioativos com potencial terapêutico para o tratamento de doenças.

[7]Via metabólica da Tiamina disponível em http://www.kegg.jp/kegg-bin/show_pathway?map00730

4. Motivação e hipótese

Atualmente, muitas informações já estão disponibilizadas em bancos de dados biológicos e encontrá-las em artigos científicos e classificá-las ainda é um desafio. Portanto, a utilização do *Text Mining* é uma abordagem computacional que pode agregar maior conhecimento às informações presentes em bancos de dados biológicos.

Esta tecnologia, também pode auxiliar quanto à classificação de artigos candidatos para complementar futuros projetos de pesquisa a partir de termos relacionados a moléculas bioativas.

Tendo como apoio a teoria de grafos, será possível visualizar os termos encontrados e as associações entre eles de acordo com os descritores disponíveis e finalmente realizar a curagem ou seleção para posterior validação ou complementação destas informações ao longo do projeto de pesquisa.

Tendo como foco de estudo as proteínas relacionadas ao virus da dengue, malária e doença de Chagas, a hipótese é que seja possível realizar a análise das associações entre os termos classificados a partir de informações obtidas em bancos de dados biológicos, para classificar se são de interesse para tratamentos terapêuticos dessas doenças.

5. Objetivos

5.1 Objetivo geral

O objetivo geral deste trabalho é prover uma abordagem computacional para viabilizar a busca, em artigos científicos, de moléculas bioativas com potencial terapêutico, de forma a estabelecer as relações que possam auxiliar na redefinição de tratamentos contra a dengue, malária e doença de Chagas.

5.2 Objetivos específicos

Para atingir o objetivo geral, identificamos os seguintes objetivos específicos:

- Desenvolver um *workflow* baseado no processo de *Text Mining* para a extração de termos a partir de bases textuais baseadas em artigos;
- Desenvolver funcionalidades para a extração das informações contidas em bases de dados sobre proteínas para validação e classificação automática dos termos extraídos das bases textuais;
- Desenvolver funcionalidades para a extração das informações contidas em bases de dados sobre estruturas de proteínas para verificar o relacionamento (e.g. similaridade estrutural) entre os termos identificados como proteínas;
- Desenvolver funcionalidades para a extração das informações contidas em bases de dados sobre fármacos para validar as proteínas como moléculas bioativas;
- Possibilitar o uso efetivo da construção de grafos como uma forma de visualização gráfica dos termos e associações entre eles.

6. Metodologia

A metodologia foi construída em duas fases complementares. A primeira relacionada à identificação e descrição das tecnologias e métodos utilizados para a busca e acesso dos artigos científicos de acordo com termos relacionados a doenças (seção 6.1). Na segunda fase estão detalhadas todas as atividades do fluxo do "WIMBAT" (Workflow para Identificação de Moléculas Bioativas em Arquivos de Texto), proposto neste trabalho para o processamento dos termos extraídos de artigos científicos (seção 6.2).

6.1 Tecnologias e métodos

6.1.1 Ambiente de desenvolvimento

Toda a metodologia foi construída tendo como ambiente de desenvolvimento o software R(67). O software R consiste em uma linguagem de programação gratuita de acordo com os termos da Licença Pública Geral GNU 3 (*General Public Licence*), criada por Ross Ihaka e Robert Gentleman na universidade de Auckland, Nova Zelândia.

Além de ser uma linguagem de programação, também é um ambiente baseado nas necessidades de computação estatística e de gráficos, bastante poderosa no que diz respeito às funcionalidades que possui e nos resultados que apresenta e inclui um conjunto extensor e em constante atualização de pacotes que ampliam a gama de funcionalidades a áreas específicas, como a mineração de textos.

Dos pacotes disponíveis no R, foram utilizados neste trabalho:

a) Hmisc (68): Desenvolvido por Frank E. Harrell Jr et al. da Universidade de Vanderbilt, contém muitas funções para o tamanho da amostra, a análise de dados, manipulação de cadeia de caracteres, dentre outras;
b) tm (69): Desenvolvido por Ingo Feinerer e Kurt Hornik. Possui um conjunto de funções para a execução de atividades de mineração de textos, como a importação de dados, manipulação, pré-processamento de textos e criação de matrizes documentos-termos;
c) RISmed (70): Possui funções que facilitam a descarga das informações presentesno bancos de dados de publicações PubMed (15) do NCBI, de forma a facilitar as análises do conteúdo deste banco de dados.

d) rJava (71): O pacote rJava permite o acesso aos métodos da biblioteca da linguagem de programação Java (72) de uma forma fácil e ágil.
e) Cluster (73): O pacote cluster possui métodos que permitem realizar atividades relacionadas à clusterização (agrupamentos) de dados para posterior análise.
f) RPostgreSQL (74): O pacote RPostgreSQL possui métodos que permitem o acesso, consulta e alteração de dados armazenados no banco de dados PostgreSQL (75)
g) XML (76): O pacote XML possui métodos que permitem visualizar e criar dados com a estrutura XML.
h) igraph (77): O pacote igraph possui métodos que permite a geração de inúmeros tipos de grafos a partir de dados informados como entrada.
i) wordcloud (78): O pacote wordcloud permite a geração de nuvem de palavras de forma a apresentar qual palavra é a mais frequente em um Corpus de textos.

6.1.2 Acesso aos artigos científicos

Para acesso aos artigos científicos, foram realizas pesquisas para a identificação de que maneira os artigos serão acessados e sobre qual filtro deveria ser criado para acessá-los de forma a atender às necessidades de uma pesquisa.

Para atingir o objetivo deste trabalho de tese foi selecionada como fonte principal para a busca dos artigos científicos o PubMed (15). Esse repositório foi escolhido por apresentar uma grande quantidade de citações para a literatura biomédica relacionada às doenças negligenciadas que atende às expectativas deste trabalho, além de ter considerável importância junto à comunidade científica.

Para a busca no PubMed podem ser utilizados filtros cuja especificidade é indicada pelo excutor do processo, fator determinante para a quantidade de artigos retornada: Suponhamos que o executor do processo necessite buscar artigos sobre dengue, independentemente de artigos que relatem trabalhos específicos sobre o vetor causador da doença ou fármacos. Para isso, ele poderia indicar apenas o termo “dengue” na busca. Caso a busca fosse mais específica (relacionada ao vetor, por exemplo), a procura poderia ser realizada mediante a utilização do termo “Aedes aegypti”.

Assim, neste trabalho foram utilizados como filtros os termos específicos sobre as doenças e um período em anos, descritos na Tabela 6.1. Especificamente, nos filtros foi indicado que os termos devem estar presentes no título dos artigos (indicado como “ti”) e os anos sendo o período em anos (indicado como “dp”).

Tabela 6.1 Filtros para a busca de artigos por doença

Doença	Filtro de Busca
Chagas	chagas[ti] OR american tripanosomiasis[ti] OR tripanosomiasis americana[ti] OR human trypanosomiasis[ti] 1960:2014[dp]
Dengue	dengue[ti] 1960:2014[dp]
Malária	malaria[ti] 1960:2014[dp]

Tal filtragem foi submetida ao PubMed através de uma função específica no pacote "RISmed" chamada `EUtilsSummary`, cujos parâmetros foram os indicados acima em momentos distintos para a criação do corpus específico a cada doença, que será descrito no item 6.2.1.

6.1.3 Validação dos termos como entidades biológicas

Para a validação dos termos foram selecionados dois repositórios de dados: UniProtKB (41)e PDB (19). O UniProtKB (41) é um banco de dados específico que faz parte de um repositório de dados de sequência de proteínas e dados de anotação chamado Uniprot (79), criado a partir de uma colaboração entre o EMBL-EBI (European Bioinformatics Institute) (80), SIB (Swiss Institute of Bioinformatics) (81) e PIR (Protein Information Resource) (82).

Nele estão armazenadas informações funcionais sobre proteínas, com anotações manuais e automáticas precisas e consistentes, por esses motivos foi selecionado como o banco de dados ideal para a realização da validação dos termos extraídos dos artigos científicos como entidades biológicas, tendo em vista ser constantemente atualizado.

Tendo as informações sobre as proteínas, tornou-se necessário também validar os termos no intuito de verificar se existem estruturas de proteínas relacionadas a eles. Para tal, foi selecionado outro conhecido banco de dados de estruturas de proteínas chamado PDB (Protein DataBank) (19).

O PDB (19) é um banco de dados criado em 1971 no Brookhaven National Laboratory (83). Nele estão armazenados os dados sobre estruturas 3D de grandes moléculas biológicas, incluindo as proteínas e ácidos nucleicos encontrados em todos os organismos, como bactérias, leveduras, plantas, moscas, de outros animais e seres humanos.

Com as informações obtidas no UniProtKB e PDB é possível ter informações que validam os termos extraídos como relacionados a uma proteína, como também obter informações sobre

em qual organismo a proteína identificada está presente, além de obter dados sobre a similaridade entre as proteínas obtidas nas validações.

6.2 Atividades do Workflow

Neste item estão descritas as atividades que fazem parte do *workflow* WIMBAT (Figura 6.1) criado para identificar em arquivos de texto possíveis proteínas com função enzimática que possam ser homólogas, inibidoras ou ativadoras de funções biológicas.

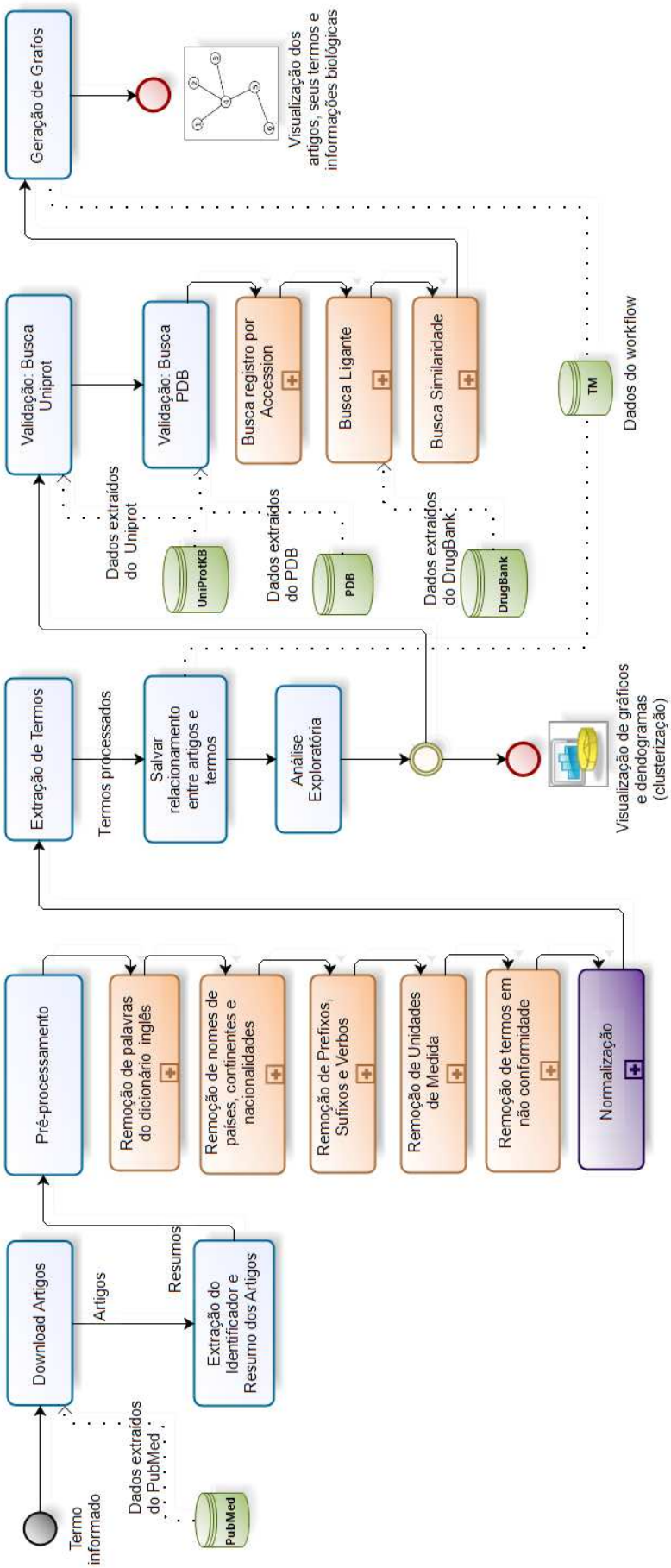

Figura 6.1*Workflow* WIMBAT para Mineração de Textos

6.2.1 Download de artigos – PubMed

Nesta atividade (Apêndice B, página 95) é realizada a seleção e download de artigos disponíveis no repositório online PubMed (15) a partir de funções disponíveis em uma biblioteca específica da linguagem R, chamada `RISmed` (70).

A partir de uma função específica desta biblioteca (`EUtilsSummary`) é realizada a criação do corpus como resultado da recuperação do conjunto de artigos, cujos títulos possuam o termo informado pelo executor do processo como filtro para busca. É possível também indicar uma quantidade máxima de artigos que podem ser recuperados. No escopo do projeto, a quantidade máxima indicada foi de 10000.

O conjunto de artigos recuperado é armazenado em uma estrutura específica da biblioteca `RISmed` contendo até este momento, apenas o identificador de cada artigo.

6.2.2 Extração do identificador e resumo dos artigos

A partir do conjunto de artigos recuperados na atividade anterior, nesta atividade é realizada a extração do conjunto de identificadores e posteriormente, a partir de outra função do `RISmed` (`EUtilsGet`) é feito o download completo do conjunto de informações específicas a cada artigo, como o título, autor(es), ano de publicação, país, ISSN, resumo, entre outros. Neste projeto, para a realização das atividades posteriores, foram recuperados o título, ano, autor(es) e resumo dos artigos, sendo este último recuperado a partir da função `AbstractText`.

Como produto principal, esta atividade gera o conjunto de textos para serem submetidos às atividades de *Text Mining*. A esse conjunto se dá o nome de “corpus”. Para a geração do corpus é utilizada uma funcionalidade (`Corpus`) de outra biblioteca do R, chamada “tm” (69).

A cada execução desta atividade é gerado um número de processo que é utilizado como identificador não só da atividade, como também dos artigos retornados por cada busca ao PubMed.

Para que todo dado obtido seja armazenado de forma a atender a metodologia e também responder a determinadas questões definidas pelo executor do processo, foi criado o banco de dados chamado “`TM`“ (Figura 6.2), construído de acordo com o sistema gerenciador de banco de dados PostgreSQL (75). Para realizar cada atualização no banco de dados “TM”, é realizada uma conexão com o SGBD PostgreSQL através da biblioteca “RPostgreSQL” (74).

Neste banco de dados, o número do processo é armazenado em uma tabela chamada "**tm.processo**", onde é depositada a data de execução da busca e o e-mail do executor do processo para identificação do mesmo.

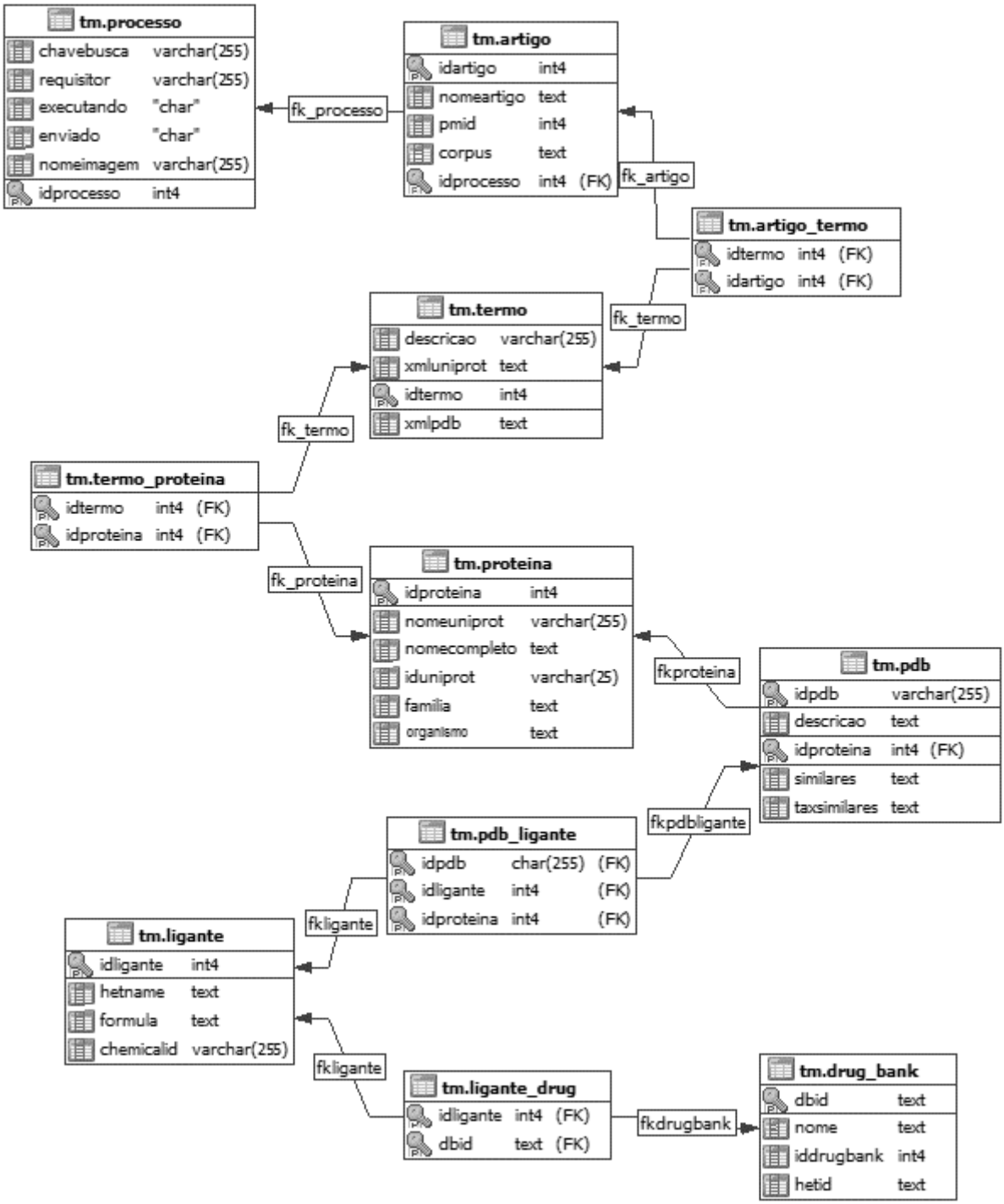

Figura 6.2 Estrutura do banco de dados de apoio à metodologia que permite em sequencia a atualização e consultas de cada tabela utilizada na execução do *workflow* (tm.processos), os termos extraídos (tm.termo, tm.artigo e tm.artigo_termo), dados sobre proteinas (tm.termo_proteina e tm.proteina), sobre a estrutura (tm.pdb), ligantes (tm.pdb_ligante e tm.ligante) e fármacos (tm.ligante_drug e tm.drug_bank)

6.2.3 Pré-processamento

O pré-processamento (Apêndice B, página 97) consta do conjunto de atividades executadas para a limpeza dos resumos, ou seja, remoção de termos que não estão de acordo com o contexto empregado neste trabalho. Para isso, são utilizadas funções relacionadas a expressões regulares.

As expressões regulares foram aplicadas em conjunto com funções específicas da linguagem R para a identificação e remoção dos termos espúrios e não interessantes nessa abordagem, por não estarem relacionados a termos candidatos a entidades biológicas (e.g proteínas, ligantes, etc). A remoção destes termos envolveu a criação de funções específicas para cada tipo de termo a ser removido.

Para a execução do pré-processamento foi necessária a criação de uma função específica chamada "`ssub`" que recebe como entrada o resumo e o conjunto de palavras que contemplam as palavras do dicionário de inglês. O pré-processamento poderia ser atendido a partir da função "`gsub`" própria da linguagem R, porém por ser uma necessidade mais complexa, a função "`ssub`" foi criada com a intenção de agilizar o processamento das exclusões das palavras.

A função "`ssub`" foi escrita utilizando comandos da linguagem C (84) que são interpretados na linguagem R, e posteriormente compilada utilizando um comando específico do R chamado SHLIB que cria um arquivo de extensão ".so" que permite a integração entre as funcionalidades da linguagem C e linguagem R.

Como retorno da função "`ssub`" tem-se uma lista de palavras que atendam o contexto deste trabalho. A seguir, serão descritas as cinco subatividades que realizam a limpeza esperada no pré-processamento:

a) Remoção de palavras do dicionário inglês: Realiza a exclusão das palavras do dicionário em inglês do resumo informado como entrada. Ao final da sua execução, temos uma lista de palavras não encontradas no dicionário de inglês mas que ainda serão validadas nas funções descritas a seguir;
b) Remoção de nomes de países/continentes/nacionalidades: Realiza a exclusão das palavras que pertencem ao conjunto de nomes de países, continentes e nacionalidades;

c) Remoção de prefixos, sufixos e verbos: Do idioma inglês que normalmente não são encontrados explicitamente no dicionário;
d) Remoção de unidades de medida: Descritas em siglas.
e) Remoção de termos em não conformidade: Compreende a remoção de palavras identificadas como não interessantes para o processo nos primeiros testes, além de números descritos em numeral e em extenso. Essas palavras formam um dicionário de termos espúrios específicos que podem ser acrescidas de acordo com a necessidade ou especificidade da extração.

6.2.4 Normalização

A normalização dos termos é uma atividade que visa complementar o pré-processamento, porém contando com algum auxílio manual para a revisão dos termos, pois alguns deles podem fazer parte do conjunto pré-processado, porém sem ter um significado que atenda ao escopo dos dados da doença submetidos ao *workflow*.

Então, com a normalização são removidos os termos que surgem repetidamente no conjunto com caixas diferentes (e.g “DENV”, “denv”, “Denv”) ou cuja escrita seja similar (e.g “denv2”, “denv-2”), sendo assim normalizados para uma mesma escrita (nos exemplos, respectivamente como “DENV” e “DENV2”).

Termos que no escopo deste trabalho é esperado que sejam encontrados a todos os artigos sobre Chagas (e.g termo “Chagas”, “cruzi”, etc), malária (e.g termo “malária”, “Falciparum”, “Falci”, “Vivax”, etc), dengue (e.g termo “dengue”) são removidos para que não sejam utilizados repetidamente em consultas ao repositório onde são validados (o UniProtKB).

6.2.5 Extração de termos

Já com o conjunto de termos que fazem parte do contexto deste trabalho, foram criadas rotinas específicas (Apêndice B, página 103) para realizar validações pontuais, para identificar se os termos poderiam fazer parte do conjunto de termos candidatos.

Tais validações incluem as descrições abaixo seguidas de exemplos de aplicação a partir da função `gsub` da linguagem R:

a) Impedir termos compostos apenas por números:

Termo validado recebe: gsub("[0-9]","",palavra)

b) Impedir termos compostos apenas por letras maiúsculas:

Termo validado recebe: gsub("[0-9]","",palavra)

c) Impedir termos compostos por alguns caracteres especiais, como ponto, hífen e formatação de matrizes (e.g 2x2):

Termo validado recebe: gsub("[0-9](x|X)[0-9]","",palavra)

d) Impedir termos compostos por codificação HTML e caracteres não ASCII (e.g & que é o sinal gráfico para apresentar o caracter “&”):

Termo validado recebe: gsub("[^\x20-\x7E]","",palavra)

e) Impedir termos compostos apenas por números e pontuações:

Termo validado recebe: gsub("[[:punct:]]|[0-9]","",palavra)

Ao final da execução desta atividade, tem-se um objeto do tipo array com três dimensões. Sendo a primeira composta pelo número do artigo (PMID); a segunda com o *corpus* do artigo; e a terceira com os termos retornados após a extração, como mostrado na Tabela 6.2.

Como resultado, o conjunto de termos que serão utilizados na próxima atividade já estará sem termos não desejados para a continuação do *workflow*.

Tabela 6.2 Exemplo do conteúdo do array retornado após a extração dos termos

Identificador	23682435
Corpus	Dengue and malaria infections are two very common vector-borne diseases annually affecting millions of people around the world. Both diseases show a variety of clinical presentations, ranging from mild symptoms of dengue fever (DF) to severe dengue hemorrhagic fever (DHF) in dengue infection, and low and high parasitemia in malaria infection. T helper (Th)1 and Th2 cytokine expressions in mild and severe forms of dengue virus type-2 (DENV-2) and Plasmodium falciparum infection, were compared to normal human sera using high throughput magnetic bead-based Bio-Plex assay. A significant elevation of Th1 and Th2 cytokines expression [interleukin (IL)-2, IL-4, IL-5, IL-10, IL-13, granulocyte-macrophage colony-stimulating factor (GM-CSF), interferon (IFN)-gamma, and tumor necrosis factor (TNF)-alpha] was detected in DENV-2 and P. falciparum malaria infections compared with normal controls ($p < 0.05$). DENV-2 infection showed a slight higher expression of Th1 and Th2 cytokines in DHF than DF, except for IL-13. In P. falciparum infection, high parasitemia showed a significantly higher expression of IL-4, IL-10, GM-CSF, and TNF-alpha ($p < 0.05$). Both DENV-2 and P. falciparum malaria infections manifested high IL-10 expression, greatest among the cytokines examined, and in the severe forms of infection. The results of this study should lead to a better understanding of pathogenesis of dengue infection and P. falciparum malaria.
Termos Extraídos	vector-borne | (DHF) | type-2 | falciparum | (GM-CSF) | DENV-2 | GM-CSF

6.2.6 Salvar relacionamento entre artigos e termos

Após a atividade de extração de termos (Apêndice B, página 106) é realizado o armazenamento dos artigos que retornaram termos candidatos para a extração de conhecimento futuro. O armazenamento dos artigos é feito através de outra funcionalidade específica, que verifica quais artigos retornaram termos candidatos. O registro desses artigos é feito na tabela **tm.artigo**.

Outra rotina nesta atividade percorre a lista que contém os termos candidatos e realiza o registro destes termos na tabela **tm.termo** e também o registro do relacionamento entre os termos candidatos e os artigos onde foram identificados na tabela **tm.artigo_termo**.

6.2.7 Análise exploratória

Ao final da remoção desses termos, teremos uma lista de termos candidatos a serem utilizados na realização das análises exploratórias, como a análise por clusterização, utilizando funções das bibliotecas `cluster` (73), `Hmisc` (68) e `MASS` (85) cuja utilização pode ser encontrada no script do Apêndice B, a partir da página 109.

Com estas bibliotecas é permitida a geração de gráficos, como dendogramas, para a realização das análises a partir de dados fornecidos. Nesta metodologia, os dados extraídos do banco de dados são transformados em uma matriz m x n de termos e documentos, onde m é composto pelo identificador dos artigos e n, os termos extraídos de todos os artigos.

Essa matriz é posteriormente convertida em um corpus de termos e documentos para a remoção de termos esparsos, que são aqueles que têm pelo menos uma percentagem escassa de elementos vazios (por exemplo, termos que ocorrem 0 (zero) vezes em um documento), através da função removeSparseTerms do pacote "`tm`".

Com o conjunto de dados já sem os termos esparsos, o mesmo é convertido para uma matriz de distância euclidiana (cujo conceito relaciona-se à medida da separação entre dois pontos, que pode ser provada pela aplicação repetida do teorema de Pitágoras, já que aplicando essa fórmula como distância, o espaço euclidiano torna-se um espaço métrico) e assim submetida à função "`hclust`" que tem como resultado os dados presentes no corpus, correlacionados de acordo com os seus graus de similaridade e transformados em uma árvore chamada dendograma, como exemplificado na Figura 6.3.

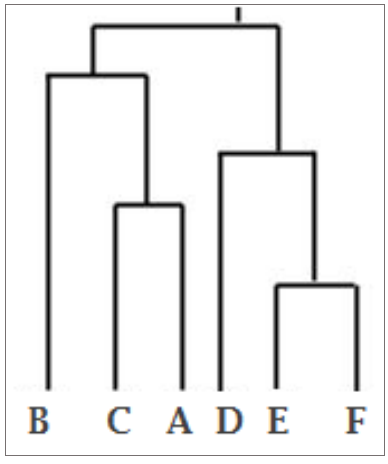

Figura 6.3 Exemplo de dendograma

Ao final da análise exploratória, fica a critério do executor continuar com a execução do *workflow* de acordo com as suas intenções, visto que algum dos resultados pode não atender

ao executor do processo, como a quantidade de artigos retornados na busca, a quantidade de termos e o contexto dos mesmos de acordo com a chave de busca utilizada como critério para busca dos artigos.

6.2.8 Validação: Busca Uniprot

A próxima atividade (Apêndice B, página 118) envolve utilizar os termos candidatos previamente registrados na tabela **tm.termo** como chave de busca para validação, ou seja, para validar se tal termo está relacionado a nomenclatura de alguma proteína ou não. Essa atividade é realizada a partir de busca textual no repositório UniProt (79).

Tal busca retorna informações de proteínas armazenadas na base de conhecimento UniProtKB (41) em formato XML (22) conforme mostrado na Figura 6.4 e, a partir dele, são extraídos os dados da proteína como o nome (tag *fullname*), número de identificação (tag *accession*) e nome do organismo onde a proteína está foi encontrada (tag *organism*).

Informações sobre a família da proteína estão presentes na tag *pfam*, porém, quando esta tag não é fornecida, as informações sobre a família são colhidas no banco de dados de famílias de proteínas chamado PFam do Instituto Sanger (86), a partir de consultas pelo número de identificação (*accession)* da proteína.

Para persistência dos dados coletados do Uniprot e o relacionamento do termo à proteína retornada na busca ao banco, nesta atividade os dados são armazenados nas tabelas **tm.proteina** e **tm.termo_proteina** respectivamente.

```
<uniprot xmlns="http://uniprot.org/uniprot" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://uniprot.org/uniprot
http://www.uniprot.org/support/docs/uniprot.xsd">
<entry dataset="Swiss-Prot" created="1987-08-13" modified="2014-02-19" version="95">
<accession>P04418</accession>
<name>END5_BPT4</name>
<protein>
<recommendedName>
<fullName>Endonuclease V</fullName>
<ecNumber>3.1.25.1</ecNumber>
</recommendedName>
</protein>
<gene>
<name type="primary">denV</name>
</gene>
<organism>...</organism>
<organismHost>...</organismHost>
<reference key="1">
<citation type="journal article" date="1984" name="Nucleic Acids
Res." volume="12" first="8085" last="8096">
<title>
Identification, physical map location and sequence of the denV gene from bacteriophage T4.
</title>
<authorList>…</authorList>
<dbReference type="PubMed" id="6095188"/>
</citation>
<scope>NUCLEOTIDE SEQUENCE [GENOMIC DNA]</scope>
</reference>
<reference key="2">...</reference>
<comment type="function">...</comment>
<dbReference type="EC" id="3.1.25.1"/></dbReference>
<proteinExistence type="evidence at protein level"/>
<keyword id="KW-0002">3D-structure</keyword>
<feature type="chain" description="Endonuclease V" id="PRO_0000164934">...</feature>
<feature type="active site" description="Proton acceptor">...</feature>
<sequence length="138" mass="16079" checksum="90B889C8E6686697" modified="1987-08-
13" version="1">
MTRINLTLVSELADQHLMAEYRELPRVFGAVRKHVANGKRVRDFKISPTFILGAGHVTFFYD
KLEFLRKRQIELIAECLKRGFNIKDTTVQDISDIPQEFRGDYIPHEASIAISQARLDE
KIAQRPTWYKYYGKAIYA
</sequence>
</entry>
<copyright>
Copyrighted by the UniProt Consortium, see http://www.uniprot.org/terms Distributed under the
Creative Commons Attribution-NoDerivs License
</copyright>
</uniprot>
```

Figura 6.4 Exemplo de informação em formato XML resultante de pesquisa no Uniprot. Adaptado de http://www.uniprot.org/uniprot/P04418.xml

6.2.9 Validação: Busca PDB

Nesta atividade, ocorre a submissão da lista de proteínas retornadas na atividade anterior a outro tipo de validação (Apêndice B, página 123), que consta da busca dos itens da lista de proteínas no banco de dados de proteínas PDB (*Protein Data Bank*) (19). Este banco de dados possui um serviço para busca de informações sobre a estrutura das proteínas chamado *RESTful Web Services*[8] que disponibiliza um conjunto de serviços para buscas no PDB a partir do tipo de dado informado.

Esta validação é dividida em duas subatividades descritas a seguir.

6.2.9.1 Busca registro por accession

Nesta subatividade é realizada a busca da estrutura a partir do número de identificação da proteína no UniProtKB (*accession*), previamente obtida na atividade anterior e recuperada na base de dados utilizada nesta metodologia.

A busca é realizada utilizando uma codificação gerada a partir da biblioteca `rJava` (71) que possui uma estrutura que permite a utilização de funções da linguagem Java (72) para o acesso e recuperação de dados a partir de chaves de busca do tipo XML.

Para cada busca no PDB, é enviada uma requisição com a estrutura XML acrescida do *accession* que identifica a proteína e o retorno é um arquivo em formato XML contendo a identificação de cada estrutura contida no banco de dados do PDB relacionada à proteína.

6.2.9.2 Busca ligantes

A partir do identificador da proteína (*accession*), nesta subatividade são obtidas as informações da estrutura da proteína, tal como o nome cuja estrutura foi batizada, sua classificação, anotações de domínio externas, enzimas homólogase informações sobre os ligantes e seus componentes químicos, além de informações destes ligantes (Apêndice B, página 128), permitindo identificar moléculas inibidoras ou ativadoras de funções biológicas relacionadas à proteína em questão.

As informações sobre os ligantes referenciam os dados disponíveis no DrugBank (87), que é um banco de dados que combina dados químicos e farmacêuticos einformações sobre alvospara tais fármacos (e.g sequência, via metabólica).

[8]RESTFul Web Services: http://www.pdb.org/pdb/software/rest.do

Ao final das duas subatividades é realizada a persistência dos dados. Os dados obtidos no PDB são armazenados na tabela **tm.pdb, tm.ligante** e **tm.pdb_ligante e** os dados coletados no DrugBank são armazenados nas tabelas**tm.drug_bank** e **tm.ligante_drug**.

6.2.10 Busca similaridade

A busca por similaridade desta subatividade (Apêndice B, página 132) consta de uma procura também realizada no PDB a partir do identificador da estrutura obtida na subatividade 6.2.9.1 (Busca registro por accession).

Agregado ao accession recuperado, a busca é realizada informando também no link para consulta o valor de corte igual a 40, ou seja, será retornado o cluster cuja similaridade seja de no mínimo 40% de forma a garantir o menor número de divergências no alinhamento utilizado para a busca de similaridade. O retorno esperado é uma lista de clusters (e.g 3JZC.A) e sua descrição (e.g para o cluster 3JZC.A tem-se “Homo sapiens”), que são concatenados e posteriormente armazenados na tabela **tm.proteina**.

6.2.11 Geração de grafos

Esta atividade (Apêndice B, página 139) envolve a apresentação das informações coletadas em forma de grafos, gerados com apoio da biblioteca `igraph` (77).

Tais gráficos são gerados visando permitir ao executor do processo a visualização dos artigos e seus termos extraídos e validados como proteínas, os artigos que estão relacionados a partir dos termos mais frequentes, além de outras formas de visualização dos dados armazenados no banco de dados, disponibilizados após a finalização da extração e validação dos termos encontrados nos artigos retornados na busca no PubMed.

7. Resultados

Os resultados refletem o retorno da execução do *workflow* proposto para a busca de artigos e realização da mineração destes artigos especificamente para dengue, malária e doença de Chagas. A execução do *workflow* foi realizada paralelamente para cada doença.

A seguir serão descritos os resultados obtidos em cada atividade do *workflow*.

7.1 Recuperação de artigos

Para cada doença foi executada a atividade que contempla o download de artigos do PubMed. (Apêndice B, página 95), onde o termo utilizado como chave de busca de artigos sobre cada doença não foi específico, para o retorno do maior número de artigos possível. Na Tabela 7.1, temos os termos utilizados como chave de busca e a quantidade de artigos retornados em 24/05/2013:

Tabela 7.1 Quantidade de artigos retornados em 24/05/2013 por chave de busca

Doença	Filtro de Busca	Recuperados do PubMed
Chagas	chagas[ti] OR american tripanosomiasis[ti] OR tripanosomiasis americana[ti] OR human trypanosomiasis[ti] 1960:2014[dp]	4762
Dengue	dengue[ti] 1960:2014[dp]	7524
Malária	malaria[ti] 1960:2014[dp]	30298

7.2 Extração da informação nos artigos

Tendo como entrada os artigos retornados na atividade anterior, foram extraídos os identificadores e resumo dos artigos (Apêndice B, página 95).

Desta forma, o resultado obtido foi uma estrutura para representar cada item do conjunto de artigos recuperados. Essa estrutura contém dentre outras informações normalmente fornecidas sobre os artigos no PubMed (e.g o(s) autor(es), jornais/periódicos e número de páginas), o título, data de publicação de cada artigo e o país, que são os dados mais relevantes para esse estudo por fazerem parte da análise exploratória a ser apresentada no item 7.4 (Análise exploratória).

7.3 Pré-processamento e extração de termos

Na atividade de extração de termos (Apêndice B, página 103) são utilizadas as expressões regulares em conjunto com funções específicas da linguagem R para a identificação e remoção dos termos espúrios, não interessantes nessa abordagem, o que foi um ponto delimitador e validador para indicar a quantidade de termos válidos para serem utilizados como entrada nas atividades seguintes.

Tabela 7.2 Quantidade de artigos recuperados do PubMed e quantidade de artigos com termos candidatos

	Quantidade de Artigos		
	Recuperados do PubMed	**Com termos candidatos**	**Porcentagem**
Chagas	4762	2770	58,16%
Dengue	7524	5150	68,5%
Malária	30298	17896	59%
Total extração	**42584**	**25816**	**60,6%**

Na Tabela 7.2 é mostrada a quantidade de artigos recuperados do PubMed e a quantidade de artigos com termos candidatos, ou seja, que foram extraídos nesta atividade e não classificados como espúrios.

Foram recuperados um total de 25816 artigos com termos candidatos, representando 60,6% dos artigos recuperados do PubMed, demonstrando uma queda do número de artigos a serem utilizados nas próximas atividades. Tal queda refere-se à presença de artigos com resumos sem termos relacionados a entidades biológicas e a presença de uma menor parte de artigos sem a descrição dos resumos.

Do total de artigos com termos candidatos identificados, 2770 artigos possuem termos relacionados a doença de Chagas, 5150 com termos relacionados à dengue e 17896 artigos com termos relacionados à malária.

Após a normalização e a validação dos termos como espúrios, a quantidade de artigos diminuiu cerca de 7% como mostra a Tabela 7.3. Portanto, o total de 13664 artigos, cujos termos agora estão validados, fez parte das próximas atividades.

Tabela 7.3 Quantidade de artigos com termos candidatos e com termos normalizados e válidos

	Quantidade de Artigos		
	Com termos candidatos	**Com termos normalizados e válidos**	**Porcentagem**
Chagas	2770	597	21,55%
Dengue	5150	1314	25,51%
Malária	17896	11753	66%
Total artigos	**25816**	**13664**	**52,92%**

A partir dos termos válidos obtidos, foi possível verificar os termos mais frequentes nos artigos relacionados às doenças que fazem parte do contexto deste trabalho. Com o pacote wordcloud (78) do ambiente R foi possível fazer essa verificação e gerar nuvens de termos (Apêndice B, página 144) contendo os termos que mais se destacam dentre o conjunto de termos de acordo com a sua frequência no conjunto.

A seguir, na Figura 7.1, é mostrado que do total de 1411 dos termos válidos obtidos sobre a doença de Chagas, utilizando como cortes para geração mínima de frequência com o valor mínimo 100 e no máximo de 200 termos, o termo mais frequente é o “benznidazol”, termo este relacionado a um fármaco criado em 1978 por Polak A. e Richle R.(88) utilizado para a quimioterapia específica da doença de Chagas.

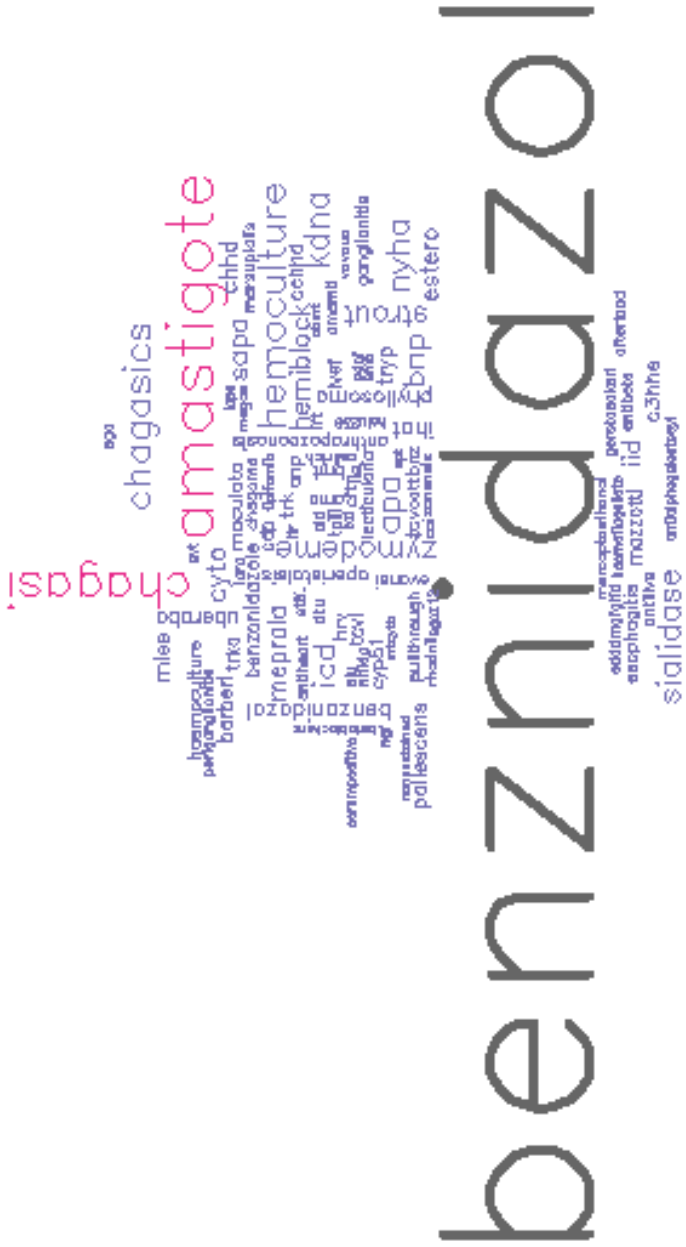

Figura 7.1 Nuvem dos termos encontrados nos artigos sobre a doença de Chagas, onde o termo mais frequente é o "benznidazol", um fármaco criado em 1978 por Polak A. e Richle R.(88) utilizado para a quimioterapia específica da doença de Chagas

Do total de 9984 dos termos válidos sobre malária, utilizando como corte para geração mínima de frequência o valor 50 e no máximo 150 termos, na Figura 7.2 é mostrado que temos três termos com maior frequência: "dox", "yoeli" e "sulfadoxin". O termo "yoeli" está relacionado ao parasita Plasmodium yoelii, uma das quatro espécies de malária que infectam roedores na África Central[9]. Os termos "dox" e "sulfadoxin" são relacionados a um fármaco utilizado no tratamento da malária, o sulfadoxin, conforme descrito por Adam e et al. em (9).

[9]Informação proveniente de http://www.genedb.org/Homepage/PyoeliiYM

Figura 7.2 Nuvem dos termos encontrados nos artigos sobre a malária, com ênfase em três termos com maior frequência: "dox", "yoeli" e "sulfadoxin". O termo "yoeli" está relacionado ao parasita Plasmodium yoelii, uma das quatro espécies de malária que infectam roedores na África Central[10]. Os termos "dox" e "sulfadoxin" são relacionados a um fármaco utilizado no tratamento da malária, o sulfadoxin

No entanto, diferentemente dos termos sobre as outras doenças, dos termos válidos relacionados aos artigos coletados sobre a dengue, somando um total de 2922 termos, na Figura 7.3 é mostrado que existe pouca diferença entre os termos mais frequentes, embora o termo "prm" relacionado à poliproteína genômica de várias proteínas relacionadas aos diferentes vírus da dengue seja o mais predominante. Foram utilizados na geração da nuvem como corte para geração mínima de frequência o valor 100 e no máximo 400 termos.

[10]Informação proveniente de http://www.genedb.org/Homepage/PyoeliiYM

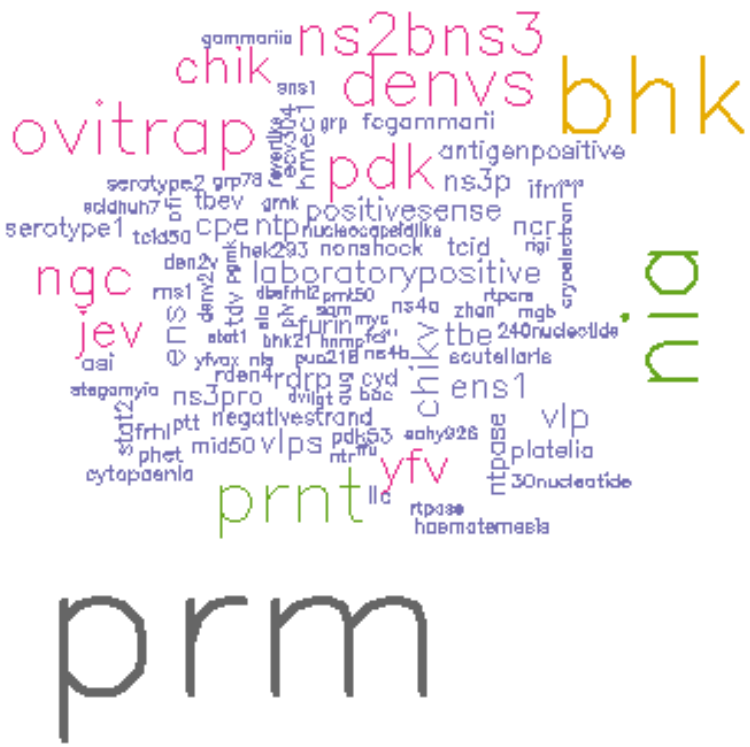

Figura 7.3 Nuvem dos termos encontrados nos artigos sobre a dengue, mostrando que existe pouca diferença entre os termos mais frequentes, embora o termo "prm" relacionado à poliproteína genômica de várias proteínas relacionadas aos diferentes vírus da dengue seja o mais predominante.

Na Tabela 7.4 estão as informações da quantidade de termos válidos por doença ao final dessa atividade, ou seja, por conjunto de artigos relacionados a cada doença. Ao total, 14317 termos foram extraídos e vários destes são referenciados pelos artigos das diferentes doenças.

Tabela 7.4 Quantidade de termos por doença (conjunto de artigos relacionado a doença)

Doença	Termos
Chagas	1411
Dengue	2922
Malária	9984
Total	**14317**

Tais termos ainda não devem ser considerados nessa fase como termos que estejam relacionados a alguma proteína, visto que ainda não foram validados a partir de consultas em banco de dados biológicos, como o Uniprot ou PDB. No entanto, parte deles pode ser utilizada como um conjunto de termos espúrios a fazerem parte desta atividade durante a execução do *workflow* em outro momento, auxiliando no refinamento do resultado do mesmo.

7.4 Análise exploratória

Na análise exploratória (Apêndice B, página 109) foram gerados gráficos para auxiliar no entendimento dos dados obtidos nas atividades anteriores.

Inicialmente foram gerados gráficos para a visualização da quantidade de artigos retornados por ano e por país para cada doença estudada, tendo como entrada de dados para a geração de cada gráfico o *corpus* retornado na busca dos artigos no PubMed.

Em seguida, foram gerados gráficos do tipo dendograma para a realização de análise de clusterização hierárquica para monitorar se os termos retornados podem ser combinados ou separados, de forma a explicar o quanto os termos estão agrupados de acordo com a presença ou não no grupo de termos extraídos de cada artigo.

Nas subseções que seguem são apresentados os gráficos relacionados aos artigos das doenças dengue, doença de Chagas e malária respectivamente.

7.4.1 Dengue

7.4.1.1 Gráficos sobre artigos e termos

Sobre a dengue foram recuperados artigos dos anos 1960 até 2013. Na Figura 7.4 é mostrada a produção de artigos por ano sobre a doença. O crescimento é exponencial desde o início das publicações, sendo o pico de produção de artigos sobre a dengue no ano de 2012, com 229 artigos publicados, o que pode aumentar até o final de 2013 se for levado em consideração o histórico.

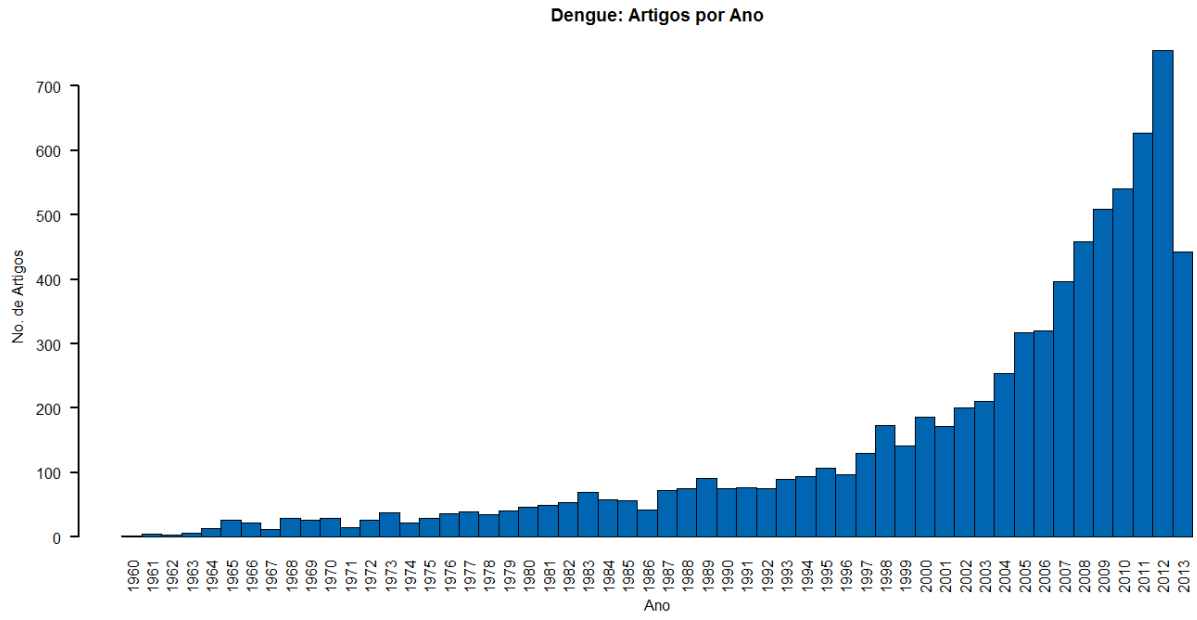

Figura 7.4 Quantidade de artigos publicados sobre a dengue de 1960 até 24/05/2013. Fonte: PubMed

7.4.1.2 Clusterização

Na Figura 7.5 é mostrada a representação de dois grupos onde o grupo da direita possui a maior quantidade de ramos (termos) semelhantes, termos esses relacionados ao vírus da dengue (e.g ns2b-ns3), amostras de RNA relacionadas ao vírus (e.g ns3p, ns3pro) ou marcadores para tratamentos (e.g pdk53), ou seja, formam um grupo mais voltado a informações quanto ao tipo e tratamento da doença.

Já à esquerda, o menor grupo da Figura 7.5, apresenta um ramo principal cujo termo (“prm”) está relacionado à poliproteína genômica de várias proteínas relacionadas aos diferentes vírus da dengue. Com esses grupos é possível verificar que os documentos selecionados para o escopo deste trabalho estão relacionados a uma grande parte de informações sobre a doença, desde os tipos da doença até o tratamento da doença em humanos.

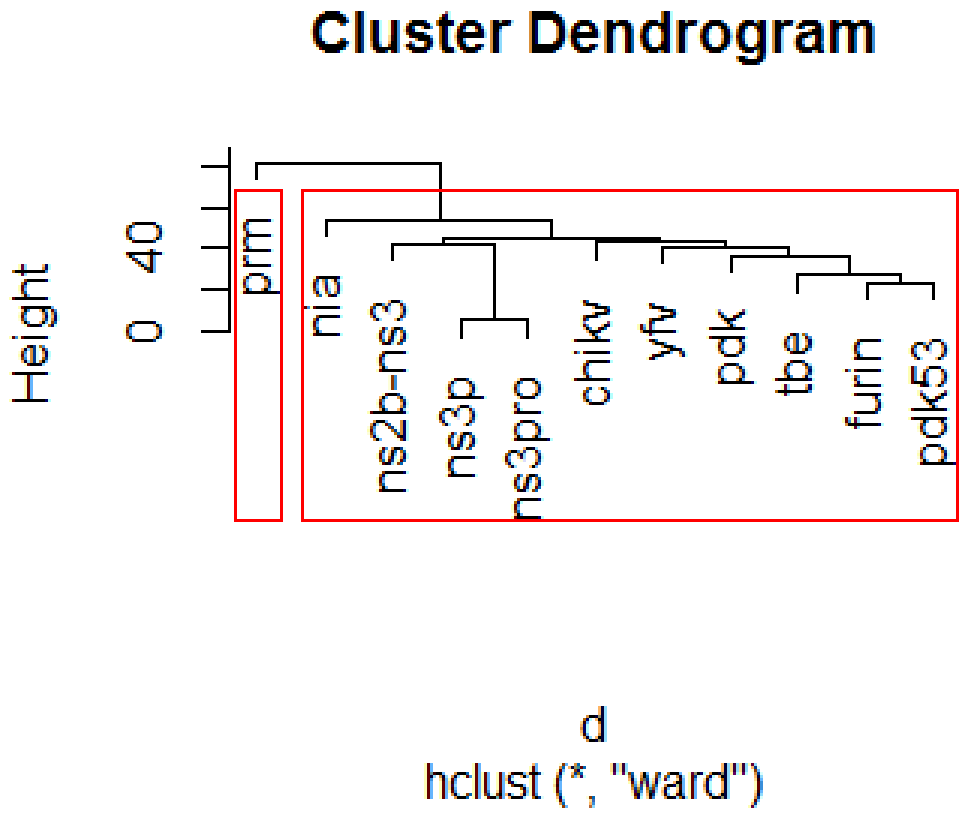

Figura 7.5 Dendograma dos grupos de dados (clusters) com os termos mais frequentes extraídos dos artigos relacionados a dengue.

7.4.2 Doença de Chagas

7.4.2.1 Gráficos sobre artigos e termos

A produção bibliográfica por ano sobre doença de Chagas como mostra a Figura 7.6, atingiu vários picos, não sendo uma produção cujo crescimento tenha sido crescente e gradual, sofrendo muita oscilação entre os anos 1960 e 2013. O maior pico de produção atingido ocorreu no ano de 2010, onde 235 artigos foram produzidos.

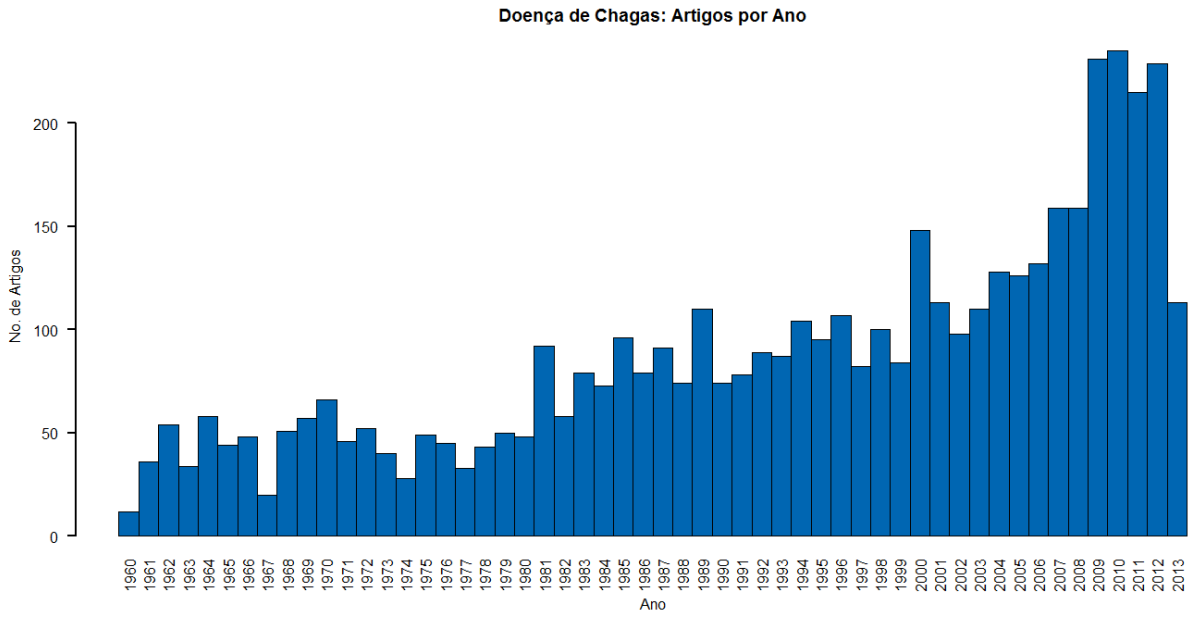

Figura 7.6 Quantidade de artigos publicados sobre Chagas de 1960 até 24/05/2013. Fonte: PubMed

7.4.2.2 Clusterização

Na Figura 7.7 é mostrado o dendograma formado por dois grupos distintos com relação ao seu significado por conta dos ramos (termos) que os pertence.

No grupo mais à esquerda, os ramos relacionados à doença de Chagas descrevem um conjunto de proteínas relacionadas a edemas inflamatórios causados pela doença de Chagas, como descrito por Scharfstein J. et al. em (60).

Já o grupo mais à direita contém termos relacionados a formas de tratamento contra a doença, como exemplo o termo “cyp51” que relaciona-se a um inibidor chamado nonazole CYP51(89). Nesse grupo também estão proteínas cujas vias metabólicas são pesquisadas para

serem inibidas, de forma a buscar a diminuição do efeito causador da doença (e.g “lanosterol” (12)) e também o grupo contém termos relacionados a receptores que atuam como porta de entrada para inflamações em tecidos (e.g “neurotrophin” é um receptor descrito em (63)).

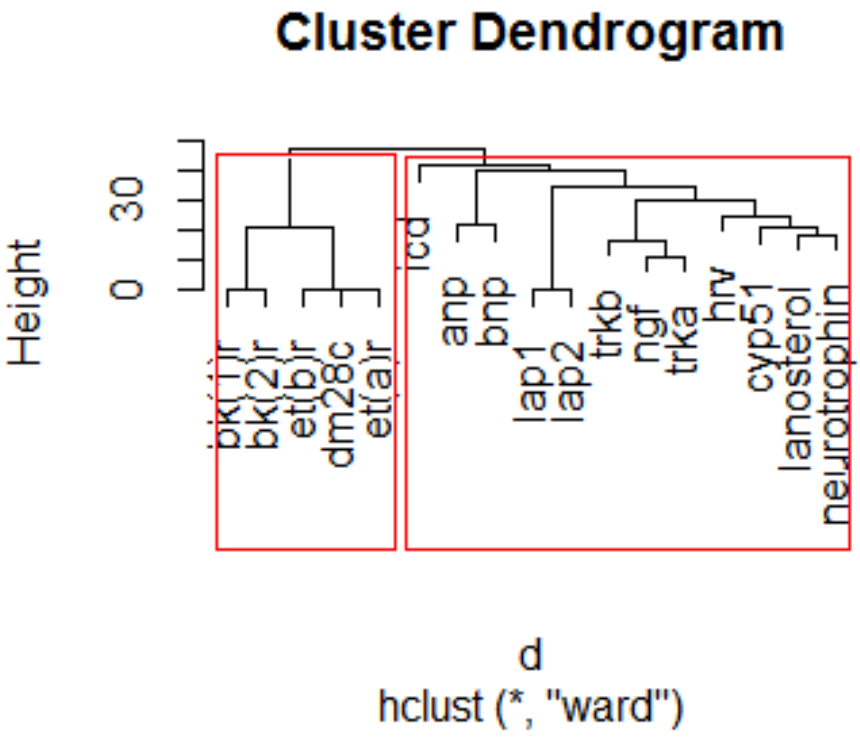

Figura 7.7 Dendograma dos grupos de dados (clusters) com os termos mais frequentes extraídos dos artigos relacionados a doença de Chagas.

Embora os grupos sejam distintos, estão no mesmo nível de altura da árvore o que significa que a maior parte dos artigos contém pelo menos dois dos termos presentes nos ramos, o que leva a indicar que as informações sobre eles descritas se complementam seja qual for o foco dos artigos, porém mantendo o escopo deste trabalho.

7.4.3 Malária

7.4.3.1 Gráficos sobre artigos e termos

Com relação à malária, foram obtidos artigos publicados por ano entre os anos de 1960 e 2013, cujo crescimento também é exponencial. O primeiro pico de produção bibliográfica ocorreu em 1969 com 237 artigos, sendo que logo após é apresentada uma diminuição nas publicações porém voltando a crescer no ano de 1981 com 267 artigos até 2012, cujo pico de produção foi de 1706 artigos conforme mostrado na Figura 7.8.

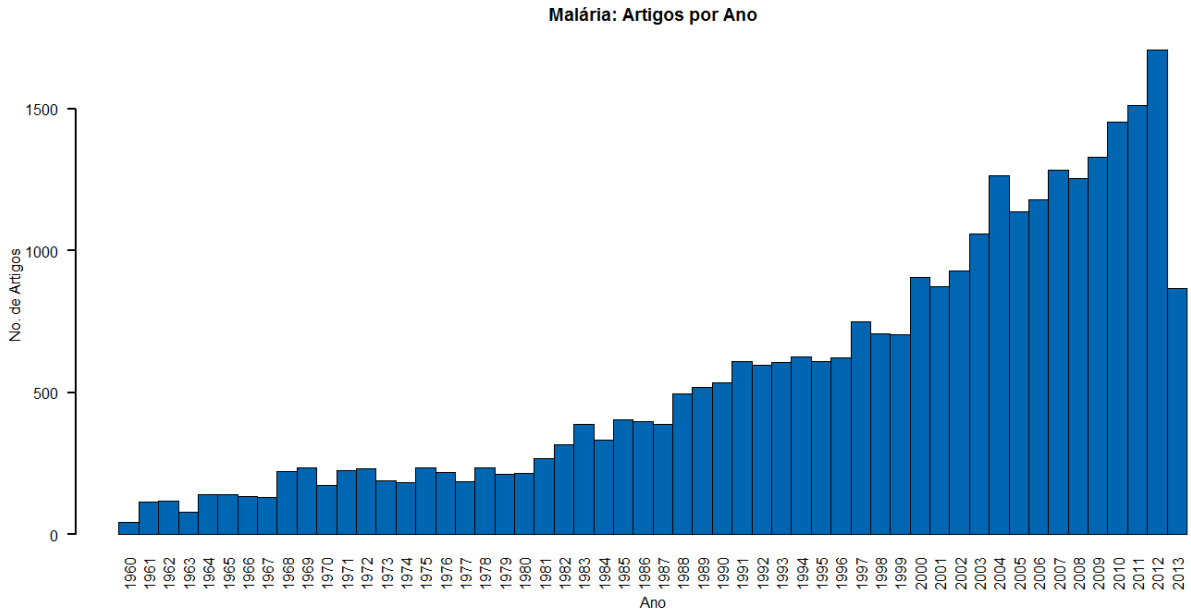

Figura 7.8 Quantidade de artigos publicados sobre a malária de 1960 até 24/05/2013. Fonte: PubMed

7.4.3.2 Clusterização

A Figura 7.9 mostra a presença de dois grupos sendo o da esquerda mais específica, formado apenas com um ramo (termo) com altura acima de 1000 no gráfico, relacionado à doxyciclina (termo “dox”), um antibiótico utilizado para conter infecções como a malária, conforme descrito no trabalho de Fall B et.al. (90).

O segundo grupo mais a direita, possui dois subgrupos que se especializam mas que se complementam em cadeia. O subgrupo mais à esquerda com termos relacionado a uma enzima presente no protozoário causador da malária (termo “cyt” cujo nome em extenso é citocromo (do inglês) “Cytochrome”). Ainda mais a direita, os termos presentes relacionam-se a proteínas (termo “dhfr”), moléculas (termo “CAM”) utilizadas em experimentos que visam fornecer novos tratamentos contra a malária, desde a diminuir a influência do parasita, quanto à criação de vacinas.

De acordo com a disposição dos ramos no gráfico, pode ser verificado que grande parte dos artigos está relacionada à descrição deformas de tratamentos contra a malária devido a estar praticamente no mesmo fator de semelhança do gráfico, indicado no vértice de altura (height), o que valida a maior parte dos termos de acordo com o contexto do trabalho.

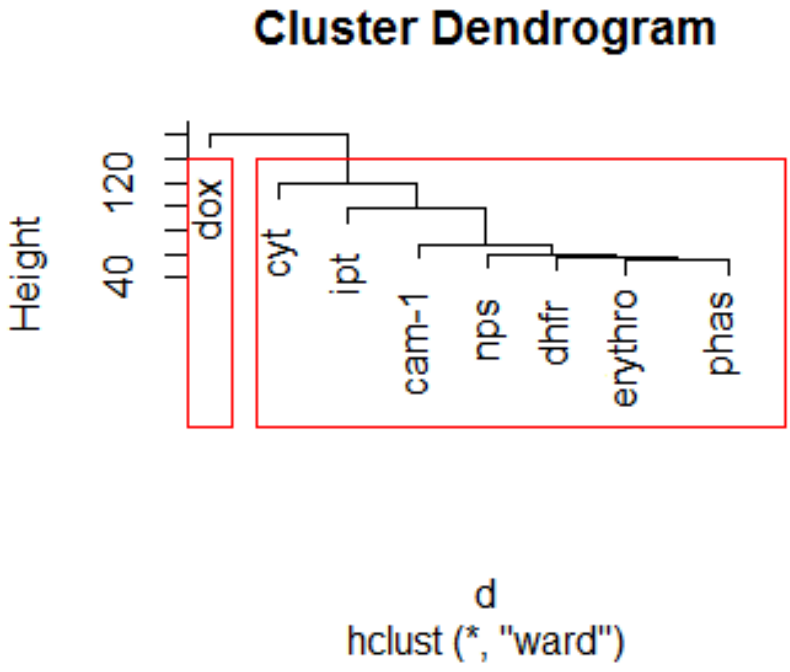

Figura 7.9 Dendograma dos grupos de dados (clusters) com os termos mais frequentes extraídos dos artigos relacionados a malária.

7.5 Salvar relacionamento entre artigos e termos

Nesta atividade, o relacionamento entre os termos e os artigos foi armazenado no banco de dados “TM” a partir de um script (Apêndice B, página 106) como previsto na metodologia.

O banco de dados, neste momento, possui preenchida toda a estrutura que serve como base para a busca nos bancos de dados biológicos, de forma que apenas os termos validados, os artigos que possuem tais termos e o relacionamento que cria a ligação entre eles são armazenados nas tabelas **tm.termo**, **tm.artigo** e **tm.artigo_termo** respectivamente.

7.6 Validação: Busca Uniprot

A validação que envolve a busca de dados no banco UniProtKB (Apêndice B, página 118) deu origem a um outro conjunto de dados: informações sobre proteínas recuperadas utilizando como chave de busca os termos validados anteriormente sendo iguais a descrição do nome da proteína ou descrição do gene.

Essas informações foram armazenadas nas tabelas **tm.proteina** e **tm.termo_proteina**. Na Tabela 7.5 é apresentada a quantidade de proteínas identificadas relacionadas a cada doença.

Tabela 7.5 Quantidade de proteínas identificadas na busca ao Uniprot por doença

Proteínas por Doença	
Dengue	23900
Chagas	11315
Malária	75280

Ao todo foram retornados 110495 itens de dado do UniProtKB relacionados aos termos extraídos, sendo um conjunto de itens relacionados a termos recuperados para os diferentes tipos de conjunto de artigos.

Além da busca ao UniProtKB, também foi necessária a busca de dados de famílias de proteínas no banco de dados de famílias PFam, para alguns itens cujo retorno da busca ao Uniprot não tinha essa informação.

7.7 Validação: Busca PDB

Nesta validação, a busca foi realizada no banco de dados de estrutura de proteínas PDB a partir do número identificador da proteína (accession), onde foram retornados 16841 identificadores de estruturas e 4844 identificadores dos ligantes relacionados a essas estruturas. Tais informações foram armazenadas nas tabelas **tm.pdb** e **tm.pdb_ligante**.

Com as informações sobre os ligantes, foi possível realizar a busca das informações sobre os fármacos no banco de dados DrugBank, que foram armazenadas nas tabelas tm.drug_bank e tm.ligante_drug. Com essa busca, foram retornados 1914 identificadores de fármacos relacionados aos ligantes das estruturas.

Na Tabela 7.6 é mostrada a quantidade de registros retornada sobre as estruturas e ligantes retornadas do PDB e fármacos retornados do DrugBank para cada doença estudada. Então, já neste momento, podemos vislumbrar que um conjunto de termos extraídos dos artigos foram validados como possíveis identificadores de moléculas bioativas, o que poderá ser visualizado a partir dos grafos apresentados no item 7.8.

Tabela 7.6 Quantidade de identificadores de estruturas, ligantes e fármacos retornados nas buscas ao PDB e Drug Bank

	Estruturas	Ligante	Fármacos
Chagas	1427	720	251
Dengue	2707	998	442
Malária	8018	2662	1090

Além das informações sobre as estruturas, ligantes e fármacos obtidos nos bancos de dados, outra informação interessante é se a proteína apresenta similaridade com quais outras proteínas.

Tal busca foi realizada através do cruzamento dos dados sobre estrutura das proteínas encontradas no PDB, visando encontrar grupos (clusters) de proteínas cuja similaridade atingisse no mínimo o valor de 40% de similaridade entre as estruturas. Com esta configuração de busca, ao todo foram retornados 20508 clusters com apontamento de similaridade relacionado a 384 organismos.

7.8 Geração de grafos

A geração de grafos é o último passo do *workflow* proposto neste trabalho. Ele permite a visualização gráfica das relações encontradas entre os dados que foram obtidos a partir dos termos extraídos dos 22074 artigos recuperados do PubMed, e validados a partir do conteúdo disponível nos bancos de dados de proteínas (UniProtKB) e estruturas de proteínas (PDB).

Os grafos foram gerados no ambiente R através do pacote `igraph` tendo como entrada as seleções na base de dados “tm” providas a partir do pacote “RPostgreSQL”.

Com os dados validados, foi possível obter informações sobre os ligantes, de forma a permitir também a visualização da relação entre os termos extraídos e dos fármacos cadastrados na base de dados DrugBank.

Nas próximas subseções serão apresentadas as diferentes formas de visualização dos dados obtidos com a execução do *workflow*, para mineração de textos científicos proposto neste trabalho, na seguinte ordem de acordo com as informações obtidas no transcorrer da execução do *workflow*: Termos relacionados às doenças, Proteínas relacionadas à família a qual descendem, Proteínas encontradas em patógenos para os humanos, Proteínas e estruturas similares, Estruturas de proteínas relacionadas ao termo, Proteínas relacionadas aos artigos coletados sobre as doenças, Ligantes relacionados aos artigos coletados sobre as doenças, Fármacos relacionados ao artigo, Fármacos relacionados às doenças, Relacionamento entre artigos através de fármacos e Proteínas de diferentes organismos relacionadas através de artigos.

7.8.1 Termos relacionados às doenças

A partir das informações coletadas nas atividades que compõem o *workflow*, foi possível obter a visualização dos termos relacionados às doenças. Com o total de 22386 termos coletados, ficaria inviável a geração de grafos contendo todos os termos relacionados às doenças. Então, foram selecionados os termos com maior frequência por doença, sendo possível construir o grafo mostrado na Figura 7.10, com frequência de corte ajustada aos limites: mínimo 5 e no máximo 7.

Nela constam os termos especificamente relacionados a artigos de apenas uma doença e termos que foram encontrados em artigos específicos sobre duas ou mais doenças.

Evidentemente, tais termos podem apenas ter sido associados a doenças diferentes por serem relacionados a metodologias de pesquisa comuns a elas, mas é interessante notar que termos identificados como presentes em organismos causadores de outras doenças, como o termo HIV (vírus da imunodeficiência humana).

Além disso, também estão relacionados à outras doenças endêmicas que estão no contexto deste trabalho. Este termo em específico possui na literatura algumas indicações sobre a associação entre essas doenças, o que já poderia levar a uma tendência na geração do grafo. Mas uma vez o mesmo ter sua construção baseada na frequência dos termos nos artigos e a chave de busca para obtenção dos artigos ter sido específica para cada doença, essa tendência torna-se praticamente nula.

Outra informação interessante que o grafo mostra é que mesmo que o termo esteja relacionado a uma doença, ele pode indiretamente levar a dados sobre outro tipo de pesquisa que não seja especificamente sobre a doença em questão, mas que pode complementar a pesquisa sobre a doença indicada para busca de artigos no *workflow* ou mesmo indicar uma direção para outra forma de pesquisa sobre a mesma.

Esse link indireto pode ser representado no grafo pelo termo “californica” que nos artigos relacionados, refere-se ao vírus de Poliedrose Nuclear Autographa californica, utilizado em pesquisas relacionadas à dengue que, por exemplo, deram origem aos artigos (91) e (92).

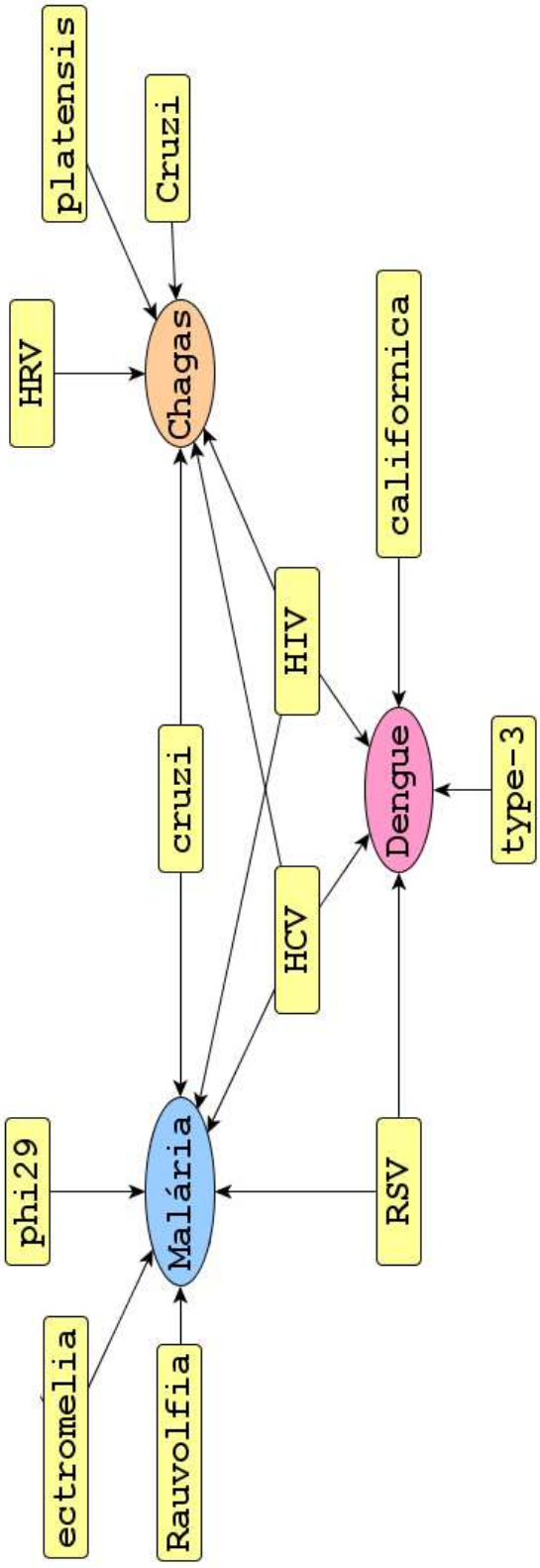

Figura 7.10 Termos mais frequentes relacionados às doenças

7.8.2 Proteínas relacionadas aos artigos coletados sobre as doenças

A partir dos dados extraídos e coletados com o *workflow*, também foi possível obter relacionamentos entre proteínas e as doenças do contexto deste trabalho. Esse relacionamento foi construído tendo como ponto de partida os termos que foram citados com maior frequência nos artigos coletados no início do *workflow*.

No grafo mostrado na Figura 7.11, as proteínas estão relacionadas a cada uma das doenças dengue, malária e Chagas, existindo também proteínas que se relacionam a mais de uma doença, o que pode indicar que tais proteínas poderiam fazer parte de pesquisas que envolvam o conhecimento indicado no relacionamento entre as doenças.

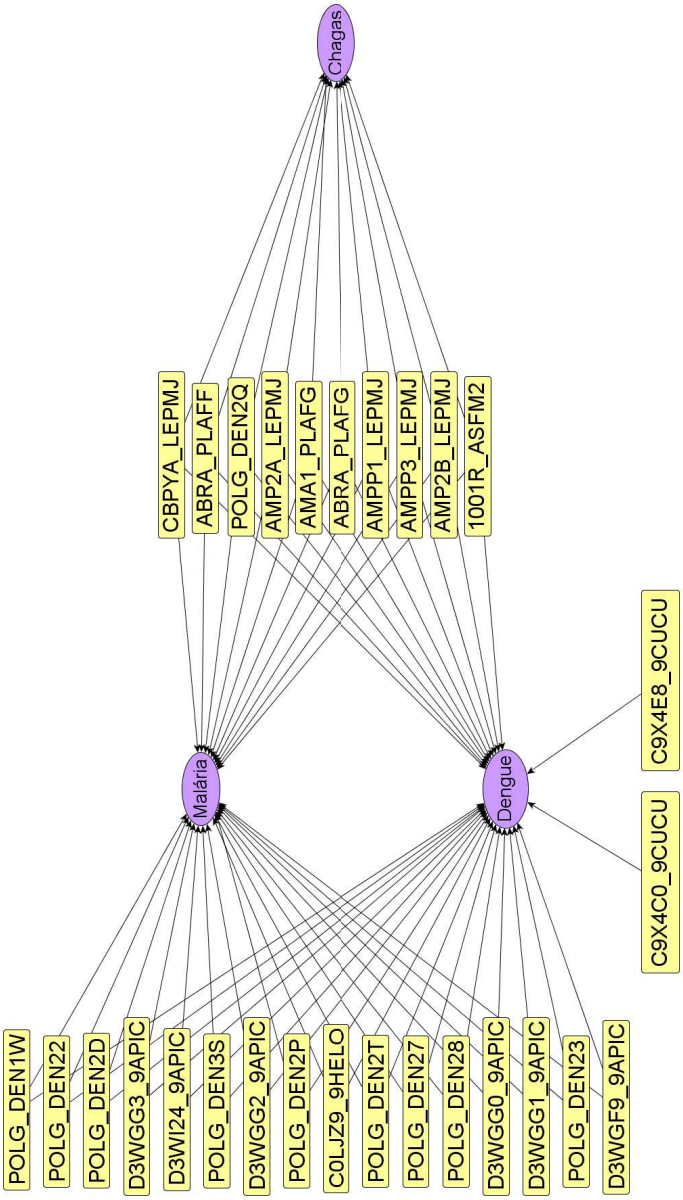

Figura 7.11 Proteínas relacionadas às doenças

7.8.3 Ligantes relacionados aos artigos coletados sobre as doenças

Como já discutido, as proteínas são moléculas cujas funções dependem de interações com outras moléculas. E com o *workflow* proposto foi possível coletar os ligantes relacionados à proteínas encontradas no UniProtKB e que se referem aos termos dos artigos coletados na primeira atividade.

Na Figura 7.12 é apresentada uma visão dos ligantes relacionados aos termos extraídos dos artigos coletados sobreas doenças dengue, malária e Chagas em específico. Assim como o grafo anterior, essas informações podem indicar que as proteínas relacionadas aos ligantes de diferentes artigos poderiam fazer parte de pesquisas que englobam duas ou mais doenças.

Neste grafo também é indicado que os ligantes podem ter interagir com diferentes proteínas presentes em organismos relacionados a doenças distintas, o que só pode ser confirmado através de experimentos em bancada molhada.

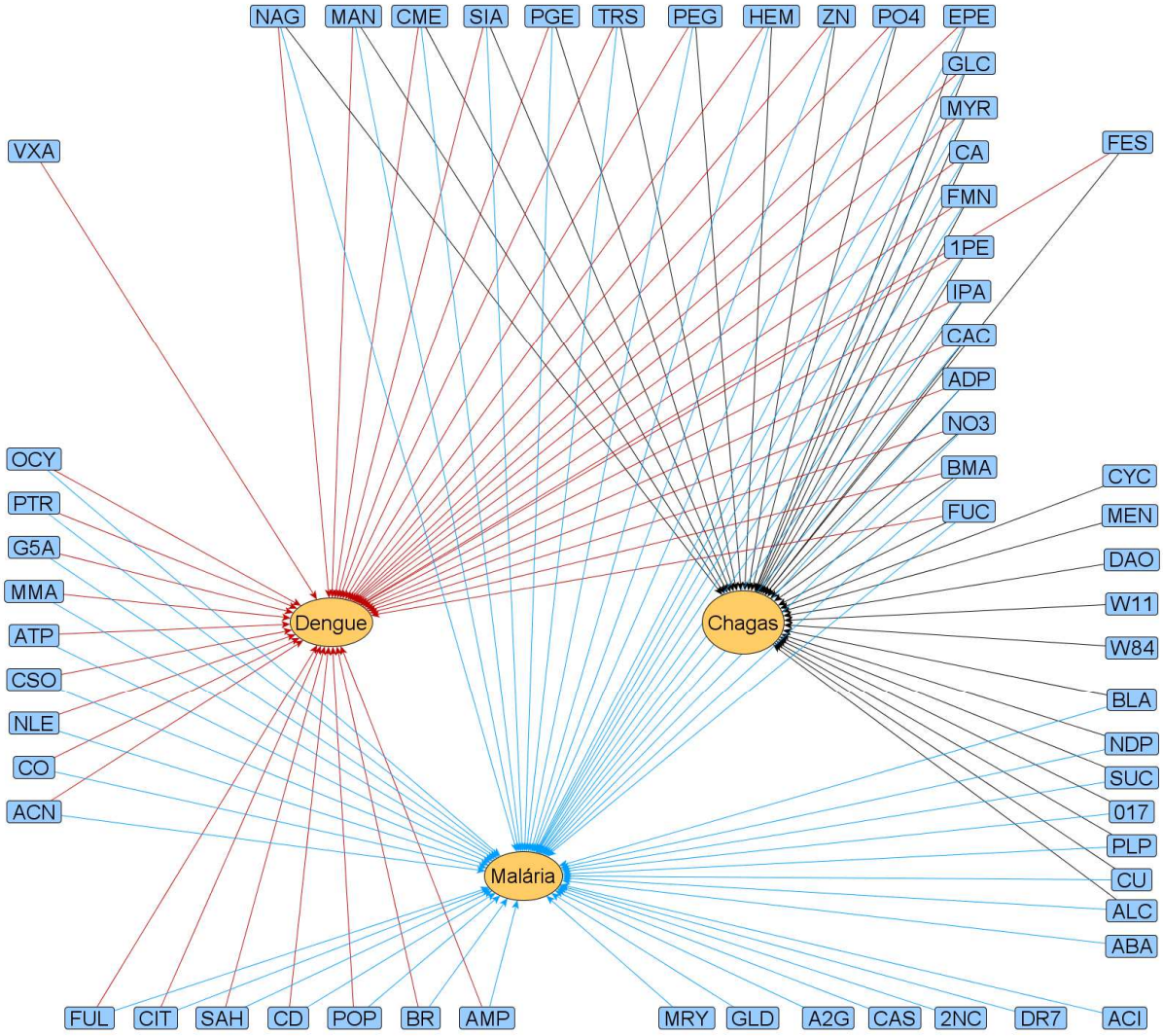

Figura 7.12 Ligantes relacionados aos artigos coletados sobre as doenças

7.8.4 Fármacos relacionados às doenças

Como o grafo anterior, a partir dos ligantes relacionados às proteínas foi possível obter as informações sobre os fármacos na base de dados do DrugBank. Desta forma, também foi possível coletar o relacionamento entre os fármacos e quais doenças estão relacionadas, ou seja, mostrar para cada doença qual fármaco foi obtido a partir de um alvo para o fármaco, a proteína, estrutura de uma proteína ou uma rota metabólica.

No grafo da Figura 7.13 são mostrados os fármacos relacionados aos artigos coletados sobre cada doença do contexto deste trabalho. A descrição dos fármacos encontra-se no Apêndice A, página 89.

Tal como as proteínas e ligantes apresentadas anteriormente, no grafo existem vários fármacos relacionados a proteínas e ligantes de mais de uma doença, podendo indicar a sua aplicação ao estudo dos fármacos a mais de uma doença.

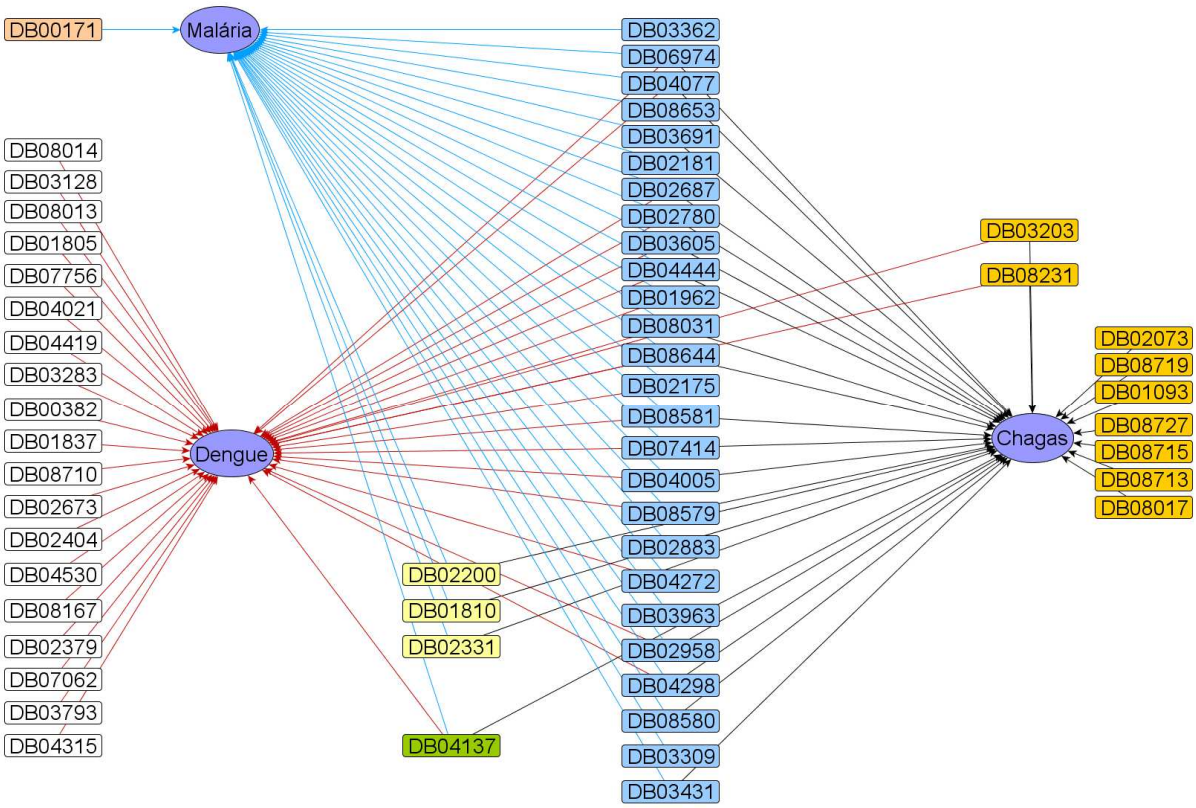

Figura 7.13 Fármacos relacionados às doenças

8. Discussão

Os levantamentos apresentados neste trabalho foram extremamente importantes para a definição da metodologia e *workflow* WIMBAT propostos, unindo conhecimento computacional e biológico obtidos durante o desenvolvimento do projeto.

Tendo em vista os dados que o *workflow* WIMBAT extraiu dos resumos dos artigos, dos 19.607 termos extraídos dos 25816 artigos como termos válidos para o escopo deste trabalho, 149.905 proteínas foram recuperadas do repositório PDB, porém deste total a quantidade de 147.102 proteínas não possuem estrutura cadastrada no PDB, o que não permitiu a busca de ligantes e consequentemente, a busca de fármacos no Drug Bank.

Na tabela presente no Apêndice A (página 82) contém uma amostra composta por 120 proteínas dos organismos *Aedes aegypti*, Dengue virus (tipos 1-4), *Plasmodium falciparum*, *Plasmodium ovale*, *Plasmodium vivax* e *Trypanosoma cruzi*, que podem ser verificados como componentes biotivos com potencial terapêutico contra as doenças relacionadas a este estudo em bancada molhada ou através de outros recursos computacionais.

Essa validação foi útil para a constatação de que ainda não fora apresentado um projeto de biologia computacional que visa auxiliar a identificação de componentes bioativos ainda não completamente explorados e que, além disso, permita ao executor do processo a visualização dos relacionamentos entre entidades biológicas de diferentes organismos e entre artigos a partir de grafos, o que pode contribuir com o avanço das pesquisas sobre essas doenças e como uma forma de diminuir custos e o tempo total dos projetos de pesquisa.

O *workflow* WIMBAT utilizou como atividade inicial a busca de termos a partir dos resumos dos artigos adquiridos no PubMed, já disponibilizados em formato texto, o que permite a extração de termos com menor esforço computacional, ou seja, sem atividades de conversão de artigos completos de PDF para formato texto.

Obviamente, utilizar o resumo ao contrário do texto completo (desde o título até a conclusão, incluindo anotações de rodapé) pode ter limitado a quantidade de termos extraídos, devido à quantidade de termos em textos completos ser maior que em resumos, consequentemente aumentando a possibilidade de serem encontrados mais termos candidatos a serem entidades biológicas.

Essa limitação já era esperada conforme descrito no trabalho de Cohen K. et al. (93), onde é apresentado que os aspectos de estrutura e conteúdo diferem acentuadamente entre resumos e texto completo de artigos. Algumas dessas diferenças podem causar problemas, corroborando com o trabalho de Lin, J. (94), onde indica que a busca em texto completo é mais eficaz do que a pesquisa resumos, especialmente quando a procura for restrita a expectativa de texto, em vez de organismos completos.

Ainda no trabalho de Cohen K. et al. (93) é indicado que essas diferenças também apresentam uma série de oportunidades para a extração de tipos de dados, especialmente a encontrada no texto entre parênteses, que está presente no corpo do artigo, mas não em resumos de artigos. Porém, para que o *Text Mining* seja realizado no texto completodo artigo, devem ser utilizados melhores analisadores, melhores formas de lidar com passivos e negação, a capacidade de lidar com o texto entre parênteses e mais atenção à detecção de uma variedade de classes semânticas além de genes e proteínas, o que foge o escopo deste trabalho.

Outra dificuldade encontrada relaciona-se a grande quantidade de siglas e abreviaturas utilizadas para nomear as entidades biológicas. No trabalho de Erhardt, R. A. et al. (12) já fora descrito que a abundância de siglas e abreviaturas é um problema adicional da literatura biomédica. Devido ao seu comprimento reduzido, estes acrônimos são frequentemente idênticos aos símbolos de genes, o que leva a aumentar a ambiguidade já existente da nomenclatura da entidade biológica e até mesmo, no caso deste projeto, a serem encontrados ambiguidades com relação a nomes de *softwares* e outras tecnologias.

Ainda em (12) é descrito que as ambigüidades e as entidades detectadas devem ser normalizadas e associadas a um objeto biológico específico de forma a diminuir essa ambiguidade. Para tal, neste projeto foi utilizado o acréscimo do nome do organismo ou doença como uma entidade de apoio à validação em cada atividade que envolvia esse problema.

Assim como as ferramentas LAITOR (55) e PESCADOR (56), o *workflow* WIMBAT procurou realizar a validação das entidades biológicas a partir de informações obtidas de dicionários pré-compilados e bancos de dados biológicos (e.g UniProtKB, PDB, DrugBank) sendo que diferentemente dessas ferramentas, este *workflow* buscou mapear o relacionamento entre as entidades a partir de ligantes e fármacos a partir das descrições obtidas em bancos de dados biológicos, tomando direção diferente à demonstração desses relacionamentos em vias

metabólicas, ou seja, apresentando ao executor do processo quais das proteínas possuem ligantes já explorados ou não como alvo para fármacos.

Como demonstrado no decorrer deste trabalho, não foi utilizada uma métrica para medir a acurácia da metodologia proposta e nem a comparação das metodologias de extração e validação de entidades biológicas a partir dos artigos científicos.

Desta forma, não foi incluído nesse projeto uma atividade detalhadamente comparativa entre as metodologias contidas na literatura que utilizam a extração de informações com o uso do Text Mining, apenas buscamos realizar uma comparação sobre o que as ferramentas oferecem e o que este projeto pode se diferenciar com relação aos objetivos e ao que se pode oferecer como resultado da extração e validação das entidades biológicas.

Com o crescimento exponencial dos artigos publicados sobre as doenças negligenciadas é notável que a necessidade de aplicação da tecnologia de *Text Mining* tem recebido maior atenção, o que leva a necessidade de ser aprimorada para permitir a visualização de relacionamentos entre entidades biológicas e artigos, sendo portanto úteis para a investigação biomédica.

Tal necessidade é indicada no trabalho de Fei Zhu et al. (12), onde também é destacado que os sistemas de *Text Mining* ainda não são as ferramentas padrão de ouro de pesquisadores biomédicos, como sistemas de recuperação e ferramentas de sequenciamento. Logo, a próxima missão importante aos sistemas de *Text Mining* é desenvolver aplicações que são realmente úteis para a investigação biomédica, onde Fei Zhu et al. indica tais necessidades em pesquisas contra o câncer.

Já com este intuito, a metodologia proposta neste trabalho foi desenvolvida e aplicada como um *workflow* cujas atividades de pré-processamento e validação se equivalem a um *plug-in*, ou seja, podendo ser ajustadas de acordo essas necessidades e também de acordo com contexto (e.g doença, organismo, etc.) indicado para análise.

9. Conclusão

Este trabalho apresentou uma metodologia desenvolvida como o *workflow* WIMBAT, desenvolvida com a linguagem R de forma genérica, ou seja, sem restrições quanto ao seu uso e quanto ao ingresso de informações para a identificação de entidades biológicas.

Com tal metodologia, construída tendo como base as atividades relacionadas à mineração de textos, foi possível demonstrar uma forma de extrair termos a partir de resumos de artigos científicos e validá-los como entidades biológicas interessantes para projetos voltados a estratégias de combate às doenças dengue, malária e Chagas, a partir do conteúdo existente em bancos de dados biológicos.

Nos próximos itens são apresentadas as contribuições e limitações sobre o trabalho desenvolvido assim como os trabalhos a serem futuramente realizados.

9.1 Lista das contribuições

Dentro deste contexto, as contribuições deste trabalho foram as que seguem:

i. Identificação de funções da biblioteca RISMed (Projeto R) como uma ferramenta que auxiliou na busca e acesso a artigos científicos de acordo com termos relacionados às doenças do escopo deste trabalho;
ii. Identificação das tecnologias e métodos para realizar a mineração de artigos científicos: onde a tecnologia selecionada é composta por bibliotecas da linguagem R que formaram o ambiente de desenvolvimento deste projeto, como a Hmisc, tm, rJava e XML, além de funções das linguagens C e Java que serviram de apoio ao ambiente;
iii. Identificação de metodologias automatizadas para validar os termos encontrados, que envolveram a utilização e construção de funções na linguagem R para o acesso aos repositórios de dados sobre proteínas (UniProtKB), estrutura de proteínas (PDB), famílias de proteínas (PFam) e fármacos (DrugBank) para validação;
iv. Criação de uma metodologia baseada em um *workflow* chamada WIMBAT para automatização do processo de busca de artigos, termos e levantamento de associação entre eles, iniciando com a busca de artigos, realizando a identificação de proteínas, suas estruturas, similaridade entre estruturas de proteínas, ligantes relacionados à

possível inibição ou ativação de funções biológicas e fármacos relacionados aos termos extraídos;

v. Com a metodologia criada foi possível identificarmos possíveis enzimas homólogas, inibidoras ou ativadoras de funções biológicas a partir dos termos identificados como entidades biológicas nos artigos utilizando funções que levaram a identificação de proteínas e posterior validação das informações obtidas sobre elas dos ligantes e estruturas similares;

vi. Foram criados grafos como parte da mesma metodologia, disponibilizando uma forma de visualização gráfica dos resultados obtidos na execução do *workflow*, permitindo a verificação dos termos extraídos, as entidades biológicas identificadas a partir deles e associações entre essas informações extraídas, ou seja, de todo conhecimento extraído a partir do proposto pela metodologia, o que pode levar a diminuir custos e o tempo total dos projetos de pesquisa;

vii. Identificou-se que até a data da busca realizada no PubMed em 24/05/2013 ainda não foi citada alguma proteína relacionada a novos fármacos para o tratamento da dengue, malária e doença de Chagas, o que indica que ainda se faz necessário mais esforços para esse tipo de pesquisa;

viii. Identificação de um número significativo de moléculas bioativas ainda não exploradas no seu potencial terapêutico, tendo em vista os dados que o *workflow* proposto extraiu dos artigos recuperados do PubMed em 24/05/2013.

9.2 Trabalhos futuros

Durante o decorrer deste trabalho, foram identificados alguns possíveis trabalhos futuros, descritos a seguir:

1. Aperfeiçoar o processo de extração, visando cobrir o contexto do termo no artigo envolvendo os adjetivos e verbos que o cercam;
2. Criação de um serviço web para disponibilizar o *workflow* WIMBAT como um serviço online disponível a qualquer momento;
3. Aplicar a metodologia em outros domínios do conhecimento, ou seja, realizar a execução do mesmo visando atender a outras doenças e outras necessidades de acordo com o domínio;

4. Estender a metodologia visando à utilização de ontologias para a validação de termos em conjunto com a validação realizada com informações extraídas de bancos de dados;
5. Criação de uma biblioteca com as atividades do *workflow* proposto para a comunidade R Project, a fim de permitir que outros colaboradores possam utilizar a ferramenta além de oferecer maior visibilidade a este projeto.

10. Referências

1. Doenças Negligenciadas. Drugs for Neglected Diseases initiative; 2010; Disponível em: http://www.dndi.org.br/pt/doencas-negligenciadas.

2. Valverde R. GlaxoSmithKline e Fiocruz ampliam colaboração para doenças negligenciadas. Agência Ficruz de Notícias. 2010.

3. Ministério da Saúde, Brasil, DCT. Doenças negligenciadas: estratégias do Ministério da Saúde. Revista de Saúde Pública. 2010;44(1): 200-202.

4. Stevens AJ, Gahan ME, Mahalingam S, Keller PA. The Medicinal Chemistry of Dengue Fever. Journal of Medicinal Chemistry. 2009;52(24):7911-26.

5. Instituto Butantan, Assessoria de Imprensa. Butantan produz vacina contra a dengue. 2011.

6. Instituto Butantan, Assessoria de Imprensa. SP começa a testar em humanos vacina contra a dengue. 2013.

7. Neto EC. Doença de Chagas - Novos conhecimentos na patogênese da Doença de Chagas. Biotecnologia Ciência & Desenvolvimento. 1999.

8. Kropf SP, Sá MR. The discovery of Trypanosoma cruzi and Chagas disease (1908-1909): tropical medicine in Brazil. História, Ciências, Saúde-Manguinhos. 2009;16:13-34.

9. Organização Mundial da Saúde, OMS. Malaria. 2013 [Acessado em: 14/08/2013]; Disponível em: http://www.who.int/mediacentre/factsheets/fs094/en/.

10. Dias RLA, Corrêa AG. Aplicações da química combinatória no desenvolvimento de fármacos. Química Nova. 2001;24:236-42.

11. Malaria Parasites. Centers for Disease Control and Prevention; Disponível em: http://www.cdc.gov/malaria/about/biology/parasites.html.

12. Zhu F, Patumcharoenpol P, Zhang C, Yang Y, Chan J, Meechai A, et al. Biomedical text mining and its applications in cancer research. Journal of Biomedical Informatics. 2013;46(2):200-11.

13. Collins WE, Barnwell JW. Plasmodium knowlesi: Finally Being Recognized. Journal of Infectious Diseases. 2009;199(8):1107-8.

14. MEDLINE. Disponível em: http://www.nlm.nih.gov/pubs/factsheets/medline.html.

15. PubMed. National Center for Biotechnology Information; Disponível em: http://www.ncbi.nlm.nih.gov/pubmed.

16. ACM. Disponível em: http://portal.acm.org.

17. IEEExplore. Disponível em: http://www.ieee.org.

18. Benson DA, Cavanaugh M, Clark K, Karsch-Mizrachi I, Lipman DJ, Ostell J, et al. GenBank. Nucleic Acids Research. 2013;41(D1):D36-D42.

19. F.C.Bernstein TFK, G.J.Williams, E.E.Meyer Jr., M.D.Brice, J.R.Rodgers, O.Kennard, T.Shimanouchi, M.Tasumi. The Protein Data Bank: A Computer-based Archival File For Macromolecular Structures. J of Mol Biol. 1977;112(3):535–42.

20. Kanehisa M, Goto S. KEGG: kyoto encyclopedia of genes and genomes. Nucleic Acids Res. 2000;28(1):27-30.

21. Morel CM, Serruya SJ, Penna GO, Guimarães R. Co-authorship Network Analysis: A Powerful Tool for Strategic Planning of Research, Development and Capacity Building Programs on Neglected Diseases. PLoS Negl Trop Dis. 2009;3(8):e501.

22. Consortium WWW. XML (eXtensible Markup Language). 1997; Disponível em: http://www.w3.org/standards/xml/.

23. Jaffe TB-LaJ. World Wide Web Consortium W3C. 1994; Disponível em: http://www.w3.org/Consortium.

24. Consortium WWW. DTD (Data Type Document). 1997; Disponível em: http://www.w3.org/XML/1998/06/xmlspec-report.htm.

25. Consortium WWW. XSD (XML Schema). 2001; Disponível em: http://www.w3.org/standards/xml/schema.

26. Lin Y, Li W, Chen K, Liu Y. A document clustering and ranking system for exploring MEDLINE citations. J Am Med Inform Assoc.14(5):651-61.

27. SALTON G, MCGILL MJ. Introduction to Modern Information Retrieval. 1983. McGraw-Hill.

28. ALVARENGA NETO RDC, MENDES KCI. Mapeamento semântico através da análise de ocorrência de descritores sobre a gestão do conhecimento. Transinformação [Internet]. 2007; 19:[19-30 pp.].

29. Srivastava A, Sahami M. Text Mining: Classification, Clustering, and Applications: Chapman \& Hall/CRC; 2009. 328 p.

30. Doms A, Schroeder M. GoPubMed: exploring PubMed with the Gene Ontology. Nucleic Acids Res. 2005;33(Web Server issue):W783-6.

31. Müller H, Kenny E, Sternberg P. Textpresso: An Ontology-Based Information Retrieval and Extraction System for Biological Literature. PLoS Biology. 2004;2:309.

32. Tan A-H. Text mining: The state of the art and the challenges. Proceedings of the Pacific Asia Conf on Knowledge Discovery and Data Mining PAKDD'99 workshop on Knowledge Discovery from Advanced Databases KDAD'99. tan99text1999. p. 65-70.

33. Witten IH, Don, Katherine J., Dewsnip, Michael e Tablan, Valentin. Text mining in a digital library. Int J Digit Libr Journal. 2004;4:56-9.

34. Witten IH, Frank E. Data Mining: Practical Machine Learning Tools and Techniques, Second Edition: Elsevier Science; 2005.

35. Goebel M, Gruenwald L. A survey of data mining and knowledge discovery software tools. SIGKDD Explor Newsl. 1999;1(1):20-33.

36. Sanchez A. Definicion e historia de los corpus. CUMBRE – Corpus Linguistico de Espanol Contemporaneo. 1995.

37. Kleene SC. Representation of events in nerve nets and finite automata. Automata Studies. 1956;34:285.

38. VantagePoint. Disponível em: http://www.thevantagepoint.com.

39. Hu X, Wu DD. Data Mining and Predictive Modeling of Biomolecular Network from Biomedical Literature Databases. IEEE/ACM Trans Comput Biol Bioinformatics. 2007;4(2):251-63.

40. Cunningham H, Wilks Y, Gaizauskas RJ. GATE: a General Architecture for Text Engineering. Proceedings of the 16th conference on Computational linguistics - Volume 2; Copenhagen, Denmark. 993365: Association for Computational Linguistics; 1996. p. 1057-60.

41. Magrane M, Consortium U. UniProt Knowledgebase: a hub of integrated protein data. Database. 2011;2011.

42. RefSeq (NCBI Reference Sequence Database). Disponível em: http://www.ncbi.nlm.nih.gov/refseq/.

43. NCBI (National Center for Biotechnology Information). Disponível em: http://www.ncbi.nlm.nih.gov/.

44. Buneman P, Cheney J, Tan W-C, Vansummeren S. Curated databases. Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems; Vancouver, Canada. 1376918: ACM; 2008. p. 1-12.

45. International Classification of Diseases. World Health Organization. 2010;10.

46. Unified Medical Language System (UMLS). 2009.

47. Logical Observation Identifiers Names and Codes (LOINC). 2012.

48. Systematized Nomenclature of Medicine (SNOMED). 2007.

49. Voet D, Voet JG, Pratt CW. Fundamentals of biochemistry: Wiley; 2000.

50. Brändén CI, Tooze J. Introduction to Protein Structure: Garland Pub.; 1999.

51. Gruber TR. Toward principles for the design of ontologies used for knowledge sharing. Int J Hum-Comput Stud. 1995;43(5-6):907-28.

52. Gene Ontology Project. 1999-2013 [Acessado em: 15/09/2013]; Disponível em: http://www.geneontology.org/.

53. Jesmin J, Jamil H. Hypothesis Driven Secondary Network Mining from PubMed Using Epiphany. 2009:483-7.

54. Al-Mubaid H, Singh RK. A New Text Mining Approach for Finding Protein-to-Disease Associations. Issn, editor2005. 145--52 p.

55. Barbosa-Silva A, Soldatos T, Magalhães I, Pavlopoulos G, Fontaine J-F, Andrade-Navarro M, et al. LAITOR - Literature Assistant for Identification of Terms co-Occurrences and Relationships. BMC bioinformatics. 2010;11(1):70.

56. Barbosa-Silva A, Fontaine J-F, Donnard E, Stussi F, Ortega M, Andrade-Navarro M. PESCADOR, a web-based tool to assist text-mining of biointeractions extracted from PubMed queries. BMC bioinformatics. 2011;12(1):435.

57. Greenacre MaH, T. The geometric interpretation of correspondence analysis. Journal of the American Statistical Association. 1987;82(398):437-47.

58. Morin A. Intensive use of factorial correspondence analysis for text mining: application with statistical education publications. Proceedings of ICOTS-7 (International Conference on Teaching Statistics). 2006.

59. Han J. Data Mining: Concepts and Techniques: Morgan Kaufmann Publishers Inc.; 2005.

60. Comaniciu D, Meer P. Mean Shift: A Robust Approach Toward Feature Space Analysis. IEEE Trans Pattern Anal Mach Intell. 2002;24(5):603-19.

61. Coulet A, Shah NH, Garten Y, Musen M, Altman RB. Using text to build semantic networks for pharmacogenomics. Journal of Biomedical Informatics.43(6):1009-19.

62. Garten Y, Altman RB. Pharmspresso: a text mining tool for extraction of pharmacogenomic concepts and relationships from full text2009. S6 p.

63. Singh S, Malik BK, Sharma DK. Molecular drug targets and structure based drug design: A holistic approach2006. 314-20 p.

64. Nelson DL. Lehninger Principles of Biochemistry: Macmillan Higher Education; 2008.

65. Moss GP. Nomenclature Committee of the International Union of Biochemistry and Molecular Biology. 1995; Disponível em: http://www.chem.qmul.ac.uk/iubmb/.

66. Niikura M, Inoue S-I, Mineo S, Yamada Y, Kaneko I, Iwanaga S, et al. Experimental cerebral malaria is suppressed by disruption of nucleoside transporter 1 but not purine nucleoside phosphorylase2013.

67. Ihaka RGaR. The R Project for Statistical Computing. 1993.

68. Frank E Harrell Jr CD. Hmisc: Harrell Miscellaneous. 2013; 3.10-1.1:[

69. Ingo Feinerer KH. tm: Text Mining Package. 2013; 0.5-8.3:[Disponível em: http://cran.r-project.org/web/packages/tm/index.html.

70. Kovalchik S. RISmed: Download content from NCBI databases. 2012; Disponível em: http://cran.r-project.org/web/packages/RISmed/index.html.

71. Urbanek S. rJava: Low-level R to Java interface. 2013 [Acessado em: 12/03/2013]; Disponível em: http://cran.r-project.org/web/packages/rJava/index.html.

72. Oracle. Java. 2013; Disponível em: http://www.oracle.com/technetwork/java/index.html.

73. Maechler M, Rousseeuw P, Struyf A, Hubert M, Hornik K. cluster: Cluster Analysis Basics and Extensions2013. Disponível em: http://cran.r-project.org/web/packages/cluster/cluster.pdf.

74. Joe Conway DE, Tomoaki Nishiyama, Sameer Kumar Prayaga, Neil Tiffin. RPostgreSQL: R interface to the PostgreSQL database system. Disponível em: http://cran.r-project.org/web/packages/RPostgreSQL/index.html.

75. Group PGD. PostgreSQL. [updated 2013-04-04Acessado em: 2013-12-05]; 9.2.4:[

76. Lang DT. Package XML2013. Disponível em: http://cran.r-project.org/web/packages/XML/XML.pdf.

77. Nepusz GCaT. The igraph software package for complex network research. InterJournal. 2006;Complex Systems:1695.

78. Fellows I. wordcloud: Word Clouds. 2013; Disponível em: http://cran.r-project.org/web/packages/wordcloud/index.html.

79. Universal Protein Resource (Uniprot). [Acessado em: 23/06/2013]; Disponível em: http://www.uniprot.org/.

80. European Bioinformatics Institute. [Acessado em: 23/06/2013]; Disponível em: http://www.ebi.ac.uk/.

81. Swiss Institute of Bioinformatics [Acessado em: 26/03/2013]; Disponível em: http://www.isb-sib.ch/.

82. Protein Information Resource [Acessado em: 23/06/2013]; Disponível em: http://pir.georgetown.edu/.

83. Brookhaven National Laboratory (BNL). [Acessado em: 23/06/2013]; Disponível em: http://www.bnl.gov/.

84. Kernighan BW, Ritchie DM. The C programming language: Prentice-Hall, Inc.; 1978. 228 p.

85. Brian Ripley BV, Kurt Hornik, Albrecht Gebhardt, David Firth. MASS: Support Functions and Datasets for Venables and Ripley's MASS. Disponível em: http://cran.r-project.org/web/packages/MASS/index.html.

86. Finn R, Mistry J, Tate J, Coggill P, Heger A, Pollington J, et al. The Pfam protein families database. Nucleic Acids Res. 2010(38(Database issue)).

87. Knox C, Law V, Jewison T, Liu P, Ly S, Frolkis A, et al. DrugBank 3.0: a comprehensive resource for 'omics' research on drugs. Nucleic Acids Res. 2011;39(Database issue):D1035-41.

88. Polak A, Richle R. Mode of action of the 2-nitroimidazole derivative benznidazole. 1978;72(1):45-54.

89. Doyle PS, Chen CK, Johnston JB, Hopkins SD, Leung SS, Jacobson MP, et al. A nonazole CYP51 inhibitor cures Chagas' disease in a mouse model of acute infection. Antimicrobial agents and chemotherapy. 2010;54(6):2480-8.

90. Fall B, Pascual A, Sarr FD, Wurtz N, Richard V, Baret E, et al. Plasmodium falciparum susceptibility to anti-malarial drugs in Dakar, Senegal, in 2010: an ex vivo and drug resistance molecular markers study. Malar J. 2013;12(1):107-.

91. Leblois H, Young PR. Maturation of the dengue-2 virus NS1 protein in insect cells: effects of downstream NS2A sequences on baculovirus-expressed gene constructs. The Journal of general virology. 1995;76 (Pt 4):979-84.

92. Deubel V, Bordier M, Megret F, Gentry MK, Schlesinger JJ, Girard M. Processing, secretion, and immunoreactivity of carboxy terminally truncated dengue-2 virus envelope proteins expressed in insect cells by recombinant baculoviruses. Virology. 1991;180(1):442-7.

93. Cohen K, Johnson H, Verspoor K, Roeder C, Hunter L. The structural and content aspects of abstracts versus bodies of full text journal articles are different. BMC Bioinformatics. 2010;11(1):1-10.

94. Lin J. Is searching full text more effective than searching abstracts? BMC Bioinformatics. 2009;10(1):1-15.

Apêndice A

Tabela de proteínas sem estrutura identificada

Organismo	Identificação Uniprot	Nome Uniprot	Nome completo da proteína	Identificação Uniprot
Aedes aegypti	Q16NS8	ANM7_AEDAE	Protein arginine N-methyltransferase 7	Q16NS8
Aedes aegypti	Q1HR36	1433Z_AEDAE	14-3-3 protein zeta	Q1HR36
Aedes aegypti	Q17GZ9	ARP5_AEDAE	Actin-related protein 5	Q17GZ9
Aedes aegypti	P49128	ACT1_AEDAE	Actin-1	P49128
Aedes aegypti	Q0IFL5	ASSY_AEDAE	Argininosuccinate synthase	Q0IFL5
Aedes aegypti	Q17PP1	AKTIP_AEDAE	Protein crossbronx homolog	Q17PP1
Aedes aegypti	Q16TM5	BND7A_AEDAE	Band 7 protein AAEL010189	Q16TM5
Aedes aegypti	O01949	ALL3_AEDAE	30 kDa salivary gland allergen Aed a 3	O01949
Aedes aegypti	P29552	ABDA_AEDAE	Homeobox protein abdominal-A homolog	P29552
Aedes aegypti	P50635	APY_AEDAE	Apyrase	P50635
Aedes aegypti	P53354	AMY1_AEDAE	Alpha-amylase I	P53354
Aedes aegypti	Q9NHW7	AQP_AEDAE	Aquaporin AQPAe.a	Q9NHW7
Aedes aegypti	Q17GS9	ARM_AEDAE	Armadillo segment polarity protein	Q17GS9
Aedes aegypti	Q0IEG8	ARP8_AEDAE	Actin-related protein 8	Q0IEG8
Aedes aegypti	Q16MG9	ASNA_AEDAE	ATPase ASNA1 homolog	Q16MG9
Aedes aegypti	Q03168	ASPP_AEDAE	Lysosomal aspartic protease	Q03168
Aedes aegypti	Q16Y34	ATAT_AEDAE	Alpha-tubulin N-acetyltransferase	Q16Y34
Aedes aegypti	B0FWC9	ATP8_AEDAE	ATP synthase protein 8	B0FWC9
Aedes aegypti	Q1HRS5	ATP6_AEDAE	ATP synthase subunit a	Q1HRS5
Dengue virus type 1 (strain	P27913	POLG_DEN1C	Genome polyprotein	P27913

Jamaica/CV1636/1977)				
Dengue virus type 1 (strain Singapore/S275/1990)	P33478	POLG_DEN1S	Genome polyprotein	P33478
Dengue virus type 1 (strain Thailand/AHF 82-80/1980)	P27912	POLG_DEN1A	Genome polyprotein	P27912
Dengue virus type 2 (isolate Malaysia M2)	P14338	POLG_DEN22	Genome polyprotein	P14338
Dengue virus type 2 (isolate Malaysia M3)	P14339	POLG_DEN23	Genome polyprotein	P14339
Dengue virus type 2 (strain China/D2-04)	P30026	POLG_DEN2D	Genome polyprotein	P30026
Dengue virus type 2 (strain Thailand/16681/1984)	P29990	POLG_DEN26	Genome polyprotein	P29990
Dengue virus type 2 (strain Thailand/TH-36/1958)	P29984	POLG_DEN2H	Genome polyprotein	P29984
Dengue virus type 3 (strain China/80-2/1980)	Q99D35	POLG_DEN3C	Genome polyprotein	Q99D35
Dengue virus type 3 (strain Martinique/1243/1999)	Q6YMS3	POLG_DEN3M	Genome polyprotein	Q6YMS3
Dengue virus type 3 (strain Sri Lanka/1266/2000)	Q6YMS4	POLG_DEN3S	Genome polyprotein	Q6YMS4
Dengue virus type 4 (strain Singapore/8976/1995)	Q5UCB8	POLG_DEN4S	Genome polyprotein	Q5UCB8
Dengue virus type 4 (strain Thailand/0476/1997)	Q2YHF2	POLG_DEN4H	Genome polyprotein	Q2YHF2
Plasmodium falciparum (isolate 3D7)	Q8I4X0	ACT1_PLAF7	Actin-1	Q8I4X0
Plasmodium	Q7KQL9	ALF_PLAF7	Fructose-	Q7KQL9

falciparum (isolate 3D7)			bisphosphate aldolase	
Plasmodium falciparum (isolate 3D7)	Q8ILW9	ACT2_PLAF7	Actin-2	Q8ILW9
Plasmodium falciparum (isolate 3D7)	Q8I5D2	ABRA_PLAF7	101 kDa malaria antigen	Q8I5D2
Plasmodium falciparum (isolate 3D7)	C6KTB7	ALTH1_PLAF7	Putative E3 ubiquitin-protein ligase protein PFF1365c	C6KTB7
Plasmodium falciparum (isolate 3D7)	Q8ILI6	AN32_PLAF7	Acidic leucine-rich nuclear phosphoprotein 32-related protein	Q8ILI6
Plasmodium falciparum (isolate 7G8)	P50492	AMA1_PLAF8	Apical membrane antigen 1	P50492
Plasmodium falciparum (isolate Camp / Malaysia)	P22620	ABRA_PLAFC	101 kDa malaria antigen	P22620
Plasmodium falciparum (isolate Camp / Malaysia)	Q99317	MSA2_PLAFC	Merozoite surface antigen 2, allelic form 1	Q99317
Plasmodium falciparum (isolate Camp / Malaysia)	P04934	MSP1_PLAFC	Merozoite surface protein 1	P04934
Plasmodium falciparum (isolate Camp / Malaysia)	P19214	EBA1_PLAFC	Erythrocyte-binding antigen 175	P19214
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P13830	RESA_PLAFF	Ring-infected erythrocyte surface antigen	P13830
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P23746	ABRA_PLAFF	101 kDa malaria antigen	P23746
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P06916	FIRA_PLAFF	300 kDa antigen AG231	P06916
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P13568	MDR_PLAFF	Multidrug resistance protein	P13568
Plasmodium falciparum (isolate	P04927	SANT_PLAFF	S-antigen protein	P04927

FC27 / Papua New Guinea)				
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P13819	MSP1_PLAFF	Merozoite surface protein 1	P13819
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P19599	MSA2_PLAFF	Merozoite surface antigen 2	P19599
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P04930	HRP_PLAFF	Small histidine-alanine-rich protein	P04930
Plasmodium falciparum (isolate FC27 / Papua New Guinea)	P13816	GARP_PLAFF	Glutamic acid-rich protein	P13816
Plasmodium falciparum (isolate fcm17 / Senegal)	P13825	ASP_PLAFS	Aspartic acid-rich protein	P13825
Plasmodium falciparum (isolate fcm17 / Senegal)	P14589	YDH3_PLAFS	Uncharacterized protein 3' to Asp-rich and His-rich proteins	P14589
Plasmodium falciparum (isolate fcm17 / Senegal)	P14587	YDH1_PLAFS	Uncharacterized protein 5' to Asp-rich and His-rich proteins	P14587
Plasmodium falciparum (isolate fcm17 / Senegal)	P14588	YDH2_PLAFS	Uncharacterized protein 3' to Asp-rich and His-rich proteins	P14588
Plasmodium falciparum (isolate fcm17 / Senegal)	P14586	HRP3_PLAFS	Histidine-rich protein	P14586
Plasmodium falciparum (isolate FCR-3 / Gambia)	P02895	GBP_PLAFG	Glycophorin-binding protein	P02895
Plasmodium falciparum (isolate FCR-3 / Gambia)	P34940	CH60_PLAFG	Chaperonin CPN60, mitochondrial	P34940
Plasmodium falciparum (isolate FCR-3 / Gambia)	P23745	ABRA_PLAFG	101 kDa malaria antigen	P23745
Plasmodium falciparum (isolate FCR-3 / Gambia)	P50490	AMA1_PLAFG	Apical membrane antigen 1	P50490
Plasmodium	Q9UAL5	ENO_PLAFG	Enolase	Q9UAL5

falciparum (isolate FCR-3 / Gambia)				
Plasmodium falciparum (isolate FCR-3 / Gambia)	P69192	SERA_PLAFG	Serine-repeat antigen protein	P69192
Plasmodium falciparum (isolate FCR-3 / Gambia)	P50649	RIR2_PLAFG	Ribonucleoside-diphosphate reductase small subunit	P50649
Plasmodium falciparum (isolate FCR-3 / Gambia)	P19260	MSA2_PLAFG	Merozoite surface antigen 2, allelic form 2	P19260
Plasmodium falciparum (isolate FCR-3 / Gambia)	P09346	KNOB_PLAFG	Knob-associated histidine-rich protein	P09346
Plasmodium falciparum (isolate FCR-3 / Gambia)	P50647	RIR1_PLAFG	Ribonucleoside-diphosphate reductase large subunit	P50647
Plasmodium falciparum (isolate fid3 / India)	P50499	MSA2_PLAFJ	Merozoite surface antigen 2	P50499
Plasmodium falciparum (isolate HB3)	P86288	ACT2_PLAFX	Actin-2	P86288
Plasmodium falciparum (isolate HB3)	P86287	ACT1_PLAFX	Actin-1	P86287
Plasmodium falciparum (isolate mad20 / Papua New Guinea)	P08569	MSP1_PLAFM	Merozoite surface protein 1	P08569
Plasmodium falciparum (isolate mad71 / Papua New Guinea)	Q03645	MSA2_PLAFZ	Merozoite surface antigen 2	Q03645
Plasmodium falciparum (isolate NF54)	P10988	ACT1_PLAFO	Actin-1	P10988
Plasmodium falciparum (isolate NF54)	P14883	ACT2_PLAFO	Actin-2	P14883
Plasmodium falciparum (isolate Nig32 / Nigeria)	Q03646	MSA2_PLAF2	Merozoite surface antigen 2	Q03646
Plasmodium falciparum (isolate Palo Alto / Uganda)	P13822	SANT_PLAFP	S-antigen protein	P13822
Plasmodium	P50495	MSP1_PLAFP	Merozoite surface	P50495

falciparum (isolate Palo Alto / Uganda)			protein 1	
Plasmodium falciparum (isolate Palo Alto / Uganda)	P20147	HSP90_PLAFP	Heat shock 90 kDa protein homolog	P20147
Plasmodium falciparum (isolate Palo Alto / Uganda)	P07765	PF2L_PLAFP	PPF2L antigen	P07765
Plasmodium falciparum (isolate Palo Alto / Uganda)	Q26005	RESA_PLAFP	Ring-infected erythrocyte surface antigen	Q26005
Plasmodium falciparum (isolate tak 9)	Q03994	MSA2_PLAF9	Merozoite surface antigen 2	Q03994
Plasmodium falciparum (isolate thtn / Thailand)	P50491	AMA1_PLAFH	Apical membrane antigen 1	P50491
Plasmodium ovale	E2CT63	E2CT63_9APIC	Cytochrome b	E2CT63
Plasmodium ovale	B3XYB0	B3XYB0_9APIC	Cytochrome b	B3XYB0
Plasmodium vivax	Q36675	COX3_PLAVI	Cytochrome c oxidase subunit 3	Q36675
Plasmodium vivax	Q9XZD6	CDC2H_PLAVI	Cell division control protein 2 homolog	Q9XZD6
Plasmodium vivax	Q0ZS46	GST_PLAVI	Glutathione S-transferase	Q0ZS46
Plasmodium vivax (strain Belem)	Q00799	RBP2_PLAVB	Reticulocyte-binding protein 2	Q00799
Plasmodium vivax (strain Belem)	P08677	CSP_PLAVB	Circumsporozoite protein	P08677
Plasmodium vivax (strain Belem)	Q00798	RBP1_PLAVB	Reticulocyte-binding protein 1	Q00798
Plasmodium vivax (strain Salvador I)	A5K5W9	ASNA_PLAVS	ATPase ASNA1 homolog	A5K5W9
Plasmodium vivax (strain Salvador I)	A5K3F9	NNRE_PLAVS	NAD(P)H-hydrate epimerase	A5K3F9
Plasmodium vivax (strain Salvador I)	A5K883	LIPA_PLAVS	Lipoyl synthase, apicoplast	A5K883
Plasmodium vivax (strain Salvador I)	A5KAL1	FEN1_PLAVS	Flap endonuclease 1	A5KAL1
Plasmodium vivax (strain Salvador I)	A5KB67	EFTS_PLAVS	Elongation factor Ts, mitochondrial	A5KB67
Plasmodium vivax (strain Salvador I)	P42666	CYSP_PLAVS	Cysteine proteinase	P42666
Plasmodium vivax (strain Salvador I)	P13826	CSP_PLAVS	Circumsporozoite protein	P13826
Plasmodium vivax (strain Salvador I)	Q9GSD3	CRT_PLAVS	Putative chloroquine resistance	Q9GSD3

			transporter	
Plasmodium vivax (strain Salvador I)	A5K6W4	ATAT_PLAVS	Alpha-tubulin N-acetyltransferase	A5K6W4
Plasmodium vivax (strain Salvador I)	O63696	CYB_PLAVS	Cytochrome b	O63696
Plasmodium vivax (strain Salvador I)	A5K5K4	DRE2_PLAVS	Anamorsin homolog	A5K5K4
Plasmodium vivax (strain Salvador I)	A5K6I6	GUF1_PLAVS	Translation factor GUF1 homolog, mitochondrial	A5K6I6
Plasmodium vivax (strain Salvador I)	A5K2F3	PURA_PLAVS	Adenylosuccinate synthetase	A5K2F3
Plasmodium vivax (strain Salvador I)	A5K4D8	COQ4_PLAVS	Ubiquinone biosynthesis protein COQ4 homolog, mitochondrial	A5K4D8
Trypanosoma cruzi	P18269	8511_TRYCR	Sialidase 85-1.1	P18269
Trypanosoma cruzi	P18271	8513_TRYCR	Sialidase 85-1.3	P18271
Trypanosoma cruzi	Q26327	DAFT_TRYCR	Trypomastigote decay-accelerating factor	Q26327
Trypanosoma cruzi	O76240	DCAM_TRYCR	S-adenosylmethionine decarboxylase proenzyme	O76240
Trypanosoma cruzi	P98023	COX2_TRYCR	Cytochrome c oxidase subunit 2	P98023
Trypanosoma cruzi	O15885	CLP_TRYCR	Heat shock protein 100	O15885
Trypanosoma cruzi	Q95046	CH60_TRYCR	Chaperonin HSP60, mitochondrial	Q95046
Trypanosoma cruzi	P18061	CALM_TRYCR	Calmodulin	P18061
Trypanosoma cruzi	P53477	ACT_TRYCR	Actin	P53477
Trypanosoma cruzi	P18270	8512_TRYCR	Sialidase 85-1.2	P18270
Trypanosoma cruzi (strain CL Brener)	Q4DTX9	ATAT2_TRYCC	Alpha-tubulin N-acetyltransferase 2	Q4DTX9
Trypanosoma cruzi (strain CL Brener)	Q4CNH2	ASNA_TRYCC	ATPase ASNA1 homolog	Q4CNH2
Trypanosoma cruzi (strain CL Brener)	Q4DLX3	COQ42_TRYCC	Ubiquinone biosynthesis protein COQ4 homolog 2, mitochondrial	Q4DLX3
Trypanosoma cruzi (strain CL Brener)	Q4CQJ5	ATAT1_TRYCC	Alpha-tubulin N-acetyltransferase 1	Q4CQJ5
Trypanosoma cruzi (strain CL Brener)	Q4DBV7	COQ41_TRYCC	Ubiquinone biosynthesis protein COQ4 homolog 1, mitochondrial	Q4DBV7

Tabela de fármacos contidos no grafo do item 7.8.4

Identificação	Nome do Fármaco	Doença
DB00114	Pyridoxal Phosphate	Malaria
DB00118	S-Adenosylmethionine	Dengue
DB00118	S-Adenosylmethionine	Malaria
DB00122	Choline	Malaria
DB00131	Adenosine monophosphate	Malaria
DB00132	Alpha-Linolenic Acid	Malaria
DB00139	Succinic acid	Dengue
DB00139	Succinic acid	Malaria
DB00141	N-Acetyl-D-glucosamine	Chagas
DB00141	N-Acetyl-D-glucosamine	Dengue
DB00141	N-Acetyl-D-glucosamine	Malaria
DB00142	L-Glutamic Acid	Chagas
DB00142	L-Glutamic Acid	Dengue
DB00142	L-Glutamic Acid	Malaria
DB00143	Glutathione	Chagas
DB00159	Icosapent	Malaria
DB00171	Adenosine triphosphate	Dengue
DB00171	Adenosine triphosphate	Malaria
DB00196	Fluconazole	Chagas
DB00196	Fluconazole	Malaria
DB00205	Pyrimethamine	Malaria
DB00290	Bleomycin	Chagas
DB00328	Indomethacin	Malaria
DB00336	Nitrofurazone	Malaria
DB00368	Norepinephrine	Dengue
DB00382	Tacrine	Dengue
DB00382	Tacrine	Malaria
DB00435	Nitric Oxide	Malaria
DB00482	Celecoxib	Malaria
DB00572	Atropine	Dengue
DB00572	Atropine	Malaria
DB00586	Diclofenac	Malaria
DB00667	Histamine Phosphate	Chagas
DB00667	Histamine Phosphate	Dengue
DB00667	Histamine Phosphate	Malaria
DB00674	Galantamine	Dengue
DB00674	Galantamine	Malaria
DB00712	Flurbiprofen	Malaria
DB00783	Estradiol	Dengue
DB00864	Tacrolimus	Malaria
DB00898	Ethanol	Chagas
DB00898	Ethanol	Dengue
DB00898	Ethanol	Malaria

DB00945	Acetylsalicylic acid	Dengue
DB00945	Acetylsalicylic acid	Malaria
DB00988	Dopamine	Dengue
DB01010	Edrophonium	Dengue
DB01010	Edrophonium	Malaria
DB01093	Dimethyl sulfoxide	Chagas
DB01093	Dimethyl sulfoxide	Dengue
DB01093	Dimethyl sulfoxide	Malaria
DB01245	Decamethonium	Dengue
DB01245	Decamethonium	Malaria
DB01643	Thymidine-5'-Phosphate	Malaria
DB01665	ZK-800270	Dengue
DB01681	Benzene Hexacarboxylic Acid	Dengue
DB01686	N,N-dimethylarginine	Chagas
DB01692	Dithioerythritol	Malaria
DB01693	Ribavirin Monophosphate	Dengue
DB01693	Ribavirin Monophosphate	Malaria
DB01694	D-tartaric acid	Chagas
DB01694	D-tartaric acid	Malaria
DB01699	(4e)-4-Aminohex-4-Enoic Acid	Malaria
DB01714	N-Methyl-Lysine	Chagas
DB01727	Isocitric Acid	Chagas
DB01747	Coprogen	Malaria
DB01752	S-Adenosyl-L-Homocysteine	Dengue
DB01752	S-Adenosyl-L-Homocysteine	Malaria
DB01756	D-4-Phosphoerythronic Acid	Malaria
DB01762	Acetoacetic Acid	Malaria
DB01769	Double Oxidized Cysteine	Chagas
DB01769	Double Oxidized Cysteine	Dengue
DB01769	Double Oxidized Cysteine	Malaria
DB01773	4-[3-Carboxymethyl-3-(4-Phosphonooxy-Benzyl)-Ureido]-4-[(3-Cyclohexyl-Propyl)-Methyl-Carbamoyl]Butyric Acid	Dengue
DB01773	4-[3-Carboxymethyl-3-(4-Phosphonooxy-Benzyl)-Ureido]-4-[(3-Cyclohexyl-Propyl)-Methyl-Carbamoyl]Butyric Acid	Malaria
DB01786	D-Alanine	Malaria
DB01803	2-(Trimethylammonium)Ethyl Thiol	Dengue
DB01803	2-(Trimethylammonium)Ethyl Thiol	Malaria
DB01805	Monoisopropylphosphorylserine	Dengue
DB01805	Monoisopropylphosphorylserine	Malaria
DB01810	[1-(1-Methyl-4,5-Dioxo-Pent-2-Enylcarbamoyl)-2-Phenyl-Ethyl]-Carbamic Acid Benzyl Ester	Chagas
DB01810	[1-(1-Methyl-4,5-Dioxo-Pent-2-Enylcarbamoyl)-2-Phenyl-Ethyl]-Carbamic Acid Benzyl Ester	Malaria
DB01812	Adenosine-3'-5'-Diphosphate	Dengue
DB01813	Pyridoxyl-Glutamic Acid-5'-Monophosphate	Malaria

DB01814	2-Tridecanoyloxy-Pentadecanoic Acid	Malaria
DB01837	O-Acetylserine	Dengue
DB01837	O-Acetylserine	Malaria
DB01863	Inositol 1,3,4,5-Tetrakisphosphate	Malaria
DB01871	[1-(1-Benzyl-3-Hydroxy-2-Oxo-Propylcarbamoyl)-2-Phenyl-Ethyl]-Carbamic Acid Benzyl Ester	Chagas
DB01871	[1-(1-Benzyl-3-Hydroxy-2-Oxo-Propylcarbamoyl)-2-Phenyl-Ethyl]-Carbamic Acid Benzyl Ester	Malaria
DB01892	Hyperforin	Dengue
DB01901	Sucrose Octasulfate	Malaria
DB01907	Nicotinamide-Adenine-Dinucleotide	Malaria
DB01910	Adenosyl-Ornithine	Dengue
DB01915	S-Hydroxycysteine	Chagas
DB01915	S-Hydroxycysteine	Malaria
DB01921	Xylose-Derived Lactam Oxime	Malaria
DB01928	Huperaine A	Dengue
DB01928	Huperaine A	Malaria
DB01942	Formic Acid	Chagas

Apêndice B

Script de funções de apoio às atividades do *workflow* WIMBAT

```
#########################################################################
#                    FUNCOES DE APOIO
#########################################################################

#########################################################################
#                    Inicio Remove Pontuacao
#########################################################################

funcaoRemovePontuacao <- function(termo) {
  tam <- nchar(termo)
  termo <- gsub("^[[:punct:]]|[[:punct:]]$","",termo)
  tamGsub <- nchar(termo)
  while (tam != tamGsub){
    tam <- nchar(termo)
    termo <- gsub("^[[:punct:]]|[[:punct:]]$","",termo)
    tamGsub <- nchar(termo)
  }
  return (termo)
}

#########################################################################
#                    Fim Remove Pontuacao
#########################################################################

#########################################################################
#                    Inicio Tratar Parenteses Chaves Conchetes
#########################################################################

funcaoTratarPaChCol <- function(termo) {
  termo <- gsub("\\(","\\\\(",termo)
  termo <- gsub("\\)","\\\\)",termo)
  termo <- gsub("\\[","\\\\[",termo)
  termo <- gsub("\\]","\\\\]",termo)
  termo <- gsub("\\{","\\\\{",termo)
  termo <- gsub("\\}","\\\\}",termo)
  termo <- gsub("\\+","\\\\+",termo)
  termo <- gsub("\\-","\\\\-",termo)
  termo <- gsub("\\*","\\\\*",termo)
  return (termo)
}

#########################################################################
#                    Fim Tratar Parenteses Chaves Conchetes
#########################################################################

#########################################################################
```

```
#                    Inicio Remover Palavras Dicionario - Banco
###############################################################################

setarDicionarioPaises <- function() {
  if (version$os == "linux-gnu"){
    endLinux <- "paisesContinentesMeses.txt"
    endDicionarioPaises <- scan(file=endLinux,sep="\t",list(""),fileEncoding="UTF-8")
  }else{
    endWindows <- "paisesContinentesMeses.txt"
    endDicionarioPaises <- scan(file=endWindows,sep="\t",list(""),fileEncoding="UTF-8")
  }
  return (endDicionarioPaises)
}

setarDicionarioPaisesPopulacao <- function() {
  if (version$os == "linux-gnu"){
    endLinux <- "paisesContinentesPopulacao.txt"
    endDicionarioPaises <- scan(file=endLinux,sep="\t",list(""),fileEncoding="UTF-8")
  }else{
    endWindows <- "paisesContinentesPopulacao.txt"
    endDicionarioPaises <- scan(file=endWindows,sep="\t",list(""),fileEncoding="UTF-8",)
  }
  return (endDicionarioPaises)
}
removerItensEspurios <- function(listaIdentificador, listaValor,listafrequencia) {

library("tm")
  library("XML")
  library("RISmed")
  library("Hmisc")
  source("scriptDatabase.r")

endDicionarioPaises <- c(setarDicionarioPaisesPopulacao(),setarDicionarioPaises())

  minusculas<-tolower(unlist(endDicionarioPaises));
  capitalizadas<-capitalize(unlist(minusculas));
  dicionario<-c(unlist(endDicionarioPaises));
  dicionario<-c(dicionario, minusculas);
  dicionario<-c(dicionario, capitalizadas);

  lendicionario <- length(dicionario)

  for(indice in 1:lendicionario){
    indiceGrep<-grep(dicionario[indice],listaValor)

    if (length(indiceGrep)>0){
      listaIdentificador<-listaIdentificador[indiceGrep*-1]
      listaValor<-listaValor[indiceGrep*-1]
      listafrequencia<-listafrequencia[indiceGrep*-1]
    }
  }
```

```
  return(list(listaIdentificador,listaValor,listafrequencia))
}

removerItensEspuriosPorRs <- function(rs,indiceComparar) {

  library("tm")
  library("XML")
  library("RISmed")
  library("Hmisc")
  source("scriptDatabase.r")

endDicionarioPaises <- c(setarDicionarioPaisesPopulacao(),setarDicionarioPaises())

  minusculas<-tolower(unlist(endDicionarioPaises));
  capitalizadas<-capitalize(unlist(minusculas));
  dicionario<-c(unlist(endDicionarioPaises));
  dicionario<-c(dicionario, minusculas);
  dicionario<-c(dicionario, capitalizadas);

  lendicionario <- length(dicionario)

for(indice in 1:lendicionario){
    indiceGrep<-grep(dicionario[indice],rs[[indiceComparar]])

    if (length(indiceGrep)>0){
      for (coluna in 1:length(rs)){
print(coluna)
        rs<-rs[[coluna]][indiceGrep*-1]
      }
    }
  }


  return(rs)

}
```

Script para atender atividade: Download Artigos – PubMed

```
##############################################################################
#               Inicio BUSCA ARTIGOS
##############################################################################

iniciarProcessoBuscaArtigo <- function(chave, requisitor, quantidade){

  # RETORNO: c(idprocesso,corpus,fetch,nomeimagem)

library("tm")
  library("XML")
  library("RISmed")
  library("Hmisc")

  source("scriptDatabase.r")

#Link com exemplos de busca de artigos
  #http://www.inside-r.org/packages/cran/RISmed/docs/EUtilsSummary
  #exemplo: dengue[ti]

  #cria o processo
  idprocesso <- armazenarProcesso(chave,requisitor)

  res <- tryCatch({

    query <- "zero"

    if (quantidade > 0){
      query <- EUtilsSummary(chave,retmax=quantidade)
}else{
      query <- EUtilsSummary(chave)
    }

  }, warning = function(war) {

    print(paste("TM_WARNING:  ",war))
    return(query)

  }, error = function(err) {

    # warning handler picks up where error was generated
    print(paste("TM_ERROR:  ",err))
    return(query)

}, finally = {

    print(query)

  })
```

```
  #retorna a quantidade de artigos retornados com a busca
  #summary(res)

  #recuperar o abstract e outros dados a partir do ID do artigo
fetch <- EUtilsGet(res)
  PMID(fetch)

#recuperar o abstract de cada paper
corpus <-Corpus(VectorSource(AbstractText(fetch)))

# Fim BUSCA ARTIGOS

  # Salva o resultado como imagem do R

  if (length(corpus)>0){

    nomeImagem <- paste("abstractsProcesso_",idprocesso,".RData",sep="");
    dados<-c(nomeImagem,idprocesso);
    salvarNomeImagem(dados);

    retorno <- list(idprocesso,corpus,fetch,nomeImagem);
    return(retorno);

  }else{

    return(0);

  }
}
```

Script para atender atividade: Pré-processamento

```
#########################################################################
#               Inicio Carrega bibliotecas
#########################################################################

library("tm")
library("XML")
library("RISmed")
library("Hmisc")

#########################################################################
#               Fim Carrega bibliotecas
#########################################################################

#########################################################################
#               Inicio CARREGA A FUNCAO SSUB - LINGUAGEM C
#########################################################################

ssub <- function(v1,v2) {
  dyn.load("vecChar3.so")
  .Call("vecChar",v1,v2)
  return(v1)
}

#########################################################################
#               Fim CARREGA A FUNCAO SSUB - LINGUAGEM C
#########################################################################



#########################################################################
#               Inicio para carregar o dicionario
#########################################################################

setarDicionario <- function() {
  #alterar para
  if (version$os == "linux-gnu"){
endLinux <- "dicionario2.txt"
    endDicionario <- scan(file=endLinux,sep=" ",list(""))
  }else{
    endWindows <- "dicionario2.txt"
    endDicionario <- scan(file=endWindows,sep=" ",list(""))
  }
  return (endDicionario)
}

endDicionario <- setarDicionario();
```

```
setarDicionarioPaises <- function() {
  if (version$os == "linux-gnu"){
    endLinux <- "paisesContinentesMeses.txt"
endDicionarioPaises <- scan(file=endLinux,sep="\t",list(""))
  }else{
    endWindows <- "paisesContinentesMeses.txt"
    endDicionarioPaises <- scan(file=endWindows,sep="\t",list(""))
}
  return (endDicionarioPaises)
}

endDicionarioPaises <- setarDicionarioPaises();

endDicionario <- c(endDicionario[[1]],endDicionarioPaises[[1]])

#############################################################################
#             Fim para carregar o dicionario
#############################################################################


#############################################################################
#             Inicio Remove as palavras carregadas
#             do dicionario
#############################################################################

removerPalavrasDoDicionario <- function(corpus, endDicionario) {

  ini= 1;
tamMax <- length(corpus);
  resp<-corpus;

minusculas<-tolower(unlist(endDicionario));
  capitalizadas<-capitalize(unlist(minusculas));
  dicionario<-c(unlist(endDicionario));
  dicionario<-c(dicionario, minusculas);
  dicionario<-c(dicionario, capitalizadas);

  retorno <- array(list(NULL), c(tamMax,1));
while (ini<=tamMax){
    if (length(corpus[[ini]]) >0){
resumo <- strsplit(corpus[[ini]]," ");
      resumo <- unlist(resumo);

      resumo <-gsub("[.]$", "",resumo)
      resumo <-gsub("[,]$", "",resumo)
      resumo <-gsub("[-]$","",resumo)
      resumo <-gsub("[.]${1}", "",resumo)
      resumo <-gsub("[,]${1}", "",resumo)
      resumo <-gsub("[-]${1}","",resumo)
      resumo <-gsub("[-]${1}","",resumo)
      resumo <-gsub("[']${1}","",resumo)
      resumo <-gsub("^[-]","",resumo)
```

```
    resumo <-gsub("^[-]$","",resumo)
    resumo <- gsub("^$"," ",resumo) #linha vazia, substitui por " " evitando erros

    lista <- ssub(resumo, dicionario);
    retorno[[ini,1]] <- as.list(lista);
   }
   ini<-ini+1;
  }

 return(retorno)
}

#############################################################################
#           Fim Remove as palavras carregadas
#           do dicionario
#############################################################################


#############################################################################
#           Inicio Remove as palavras específicas
#           (dicionário próprio) e algumas palavras do dicionário
#           de inglês
#############################################################################

removeCInvalidos <- function(corpus) {

d <- c("RNA", "DNA", "ELISA","IMPORTANCE","MUTATION","CELL","CELLS",
       "CONCLUSION", "OBJECTIVE", "&QUOT","ABSTRACT","KIT","KITS",
       "TOOLS","TOOL", "NUCLEUS", "CULTURE", "VACCINE", "VIVO",
       "COLI", "HUMAN", "HUMANS", "STRUCTURE", "RATE", "DOMAIN",
"DOMAINS","INTRA","-INTRA","INTRA-","APOPTOSIS","ELISAS",
"(ELISA)", "BACKGROUND", "METHODS", "METHOD", "NILE", "ETC",
"ADULT", "STEM", "LOOP", "NOVO", "ROSETTA","I.C")

x <-gsub("^\\s","",corpus)
 x <-gsub("\\s$","",x)
 x <-gsub("[.]$", "",x)
 x <-gsub("[:]$", "",x)
 x <-gsub("[;]$", "",x)
 x <-gsub("[,]$", "",x)
 x <- strsplit(x," ");
 x <- unlist(x);
 x <- ssub(x, d);
 x <- ssub(x, tolower(d))
 x <- ssub(x, capitalize(tolower(d)))
return (x)
}

#############################################################################
#           Fim Remove as palavras especificas
#           (dicionario próprio) e algumas palavras do dicionario
#           de inglês
```

```
#########################################################################


#########################################################################
#                    Inicio Removedor de Substantivos,
#                         verbos e adjetivos
#########################################################################

removeSubstantivosVerbos <- function(palavra) {
  tamInicial<-nchar(palavra);
  resposta <- palavra;
  maiuscula <- toupper(palavra);
  ### Sufixos mapeados
  if (nchar(gsub("[A-
        Z](TY|IONS|IES|RAL|ENTS|ACT|ACTS|CY|OPE|FIC|EX|UDY|ATES|NESIS|SON|U
        RES|TIN|ASIS|IDS|TS|US|-LIKE|ES|ANS|EL|ELS|VIR|'S|ANT)+$","",maiuscula))!=
        tamInicial){
    resposta = ""
  }else if (nchar(gsub("[0-9](-DOMAIN|-HUMAN|-FOLD|-INTRA|-ELISA|&QUOT|-
        KDA)|[A-Z](-DOMAIN|-HUMAN|-FOLD|-INTRA|-ELISA|&QUOT|-
        KDA)+$","",maiuscula))!= tamInicial){
    resposta = ""
  }else if (nchar(gsub("[0-9](MONTH|YEAR|DAY|KHZ|PPM|LOG|MIN|WEEK|OLD)|[A-
        Z](MONTH|YEAR|DAY|KHZ|PPM|LOG|MIN|WEEK|OLD)|[[:punct:]](MONTH|YE
        AR|DAY|KHZ|PPM|LOG|MIN|WEEK|OLD)+$","",maiuscula))!= tamInicial){
    resposta = ""
  }
  ### Sufixos ingles
  else if (nchar(gsub("[A-
        Z](ER|OR|AR|IST|INA|IST|IAN|ION|ATION|ITION|MENT|ANCE|ENCE|AL|AGE|H
        OOD|SHIP|DOM|ERY|ING|FUL|NESS|ITY|FUL|LESS|ABLE|IBLE|Y|LY|IVE|WOR
        THY|OUS|ED|ING|IC|IFY|IZE|ISE|EN)+$","",maiuscula))!= tamInicial){
    resposta = ""
  }
  ### Prefixos variados
  else if (nchar(gsub("(ELISA-|ELISA|PRIME-
        |UN|IN|IL|IM|IR|DIS|NON|UN|DE|DIS|MIS|MAL|PSEUDO|ARCH|SUPER|OUT|SU
        R|SUB|OVER|UNDER|HYPER|ULTRA|MINI|CO|COUNTER|ANTI|PRO|SUPER|S
        UB|INTER|TRANS|FORE|PRE|POST|EX|RE|UNI|MONO|BI|DI|TRI|MULTI|POLY|
        AUTO|NEO|PAN|PROTO|SEMI|VICE|INTRA-)[A-Z]{4,}","",maiuscula))!=
        tamInicial){
    resposta = ""
  } else  if (nchar(gsub("[0-9]%","",maiuscula))!= tamInicial){
    resposta = "";
} else  if (nchar(gsub("=","",maiuscula))!= tamInicial){
    resposta = "";
  } else  if (nchar(gsub("/HOUR|/HUMAN|/WEEK|-WEEK","",maiuscula))!= tamInicial){
    resposta = "";
  } else  if (nchar(gsub("[0-9]", "", maiuscula," ")) == 0){
    resposta = "";
  } else if (nchar(gsub("(ELISA-|ELISA|PRIME-
        |UN|IN|IL|IM|IR|DIS|NON|UN|DIS|MIS|MAL|PSEUDO|ARCH|SUPER|OUT|SUR|S
```

```
        UB|OVER|UNDER|HYPER|ULTRA|MINI|CO|COUNTER|ANTI|PRO|SUPER|SUB|I
        NTER|TRANS|FORE|PRE|POST|EX|RE|UNI|MONO|BI|DI|TRI|MULTI|POLY|AUT
        O|NEO|PAN|PROTO|SEMI|VICE|INTRA-)[A-Z]","",maiuscula))!= tamInicial){
resposta = ""
  }

  return (resposta)
}

#############################################################################
#                    Fim Removedor de Substantivos,
#                         verbos e adjetivos
#############################################################################

#############################################################################
#                    Inicio Removedor de Unidades de Medida
#############################################################################

removeUnidadesMedida <- function(palavra) {

  resposta <- palavra;
  if (length(palavra)>0){

    tamInicial<-nchar(palavra);
    maiuscula <- toupper(palavra);

    if
        (nchar(gsub("[/](G|DAG|HG|KG|MG|GG|TG|PG|EG|ZG|YG|DG|CG|MG|ÂµG|NG|PG
        |FG|AG|ZG|YG|MOL)|[0-
        9](G|DAG|HG|KG|MG|GG|TG|PG|EG|ZG|YG|DG|CG|MG|ÂµG|NG|PG|FG|AG|ZG|Y
        G|MOL)","",maiuscula))!= tamInicial){
      resposta = ""
    }else if
        (nchar(gsub("[/](M|DAM|HM|KM|MM|GM|TM|PM|EM|ZM|YM|DM|CM|MM|ÂµM|
        NM|PM|FM|AM|ZM|YM|MOL)|[0-
        9](M|DAM|HM|KM|MM|GM|TM|PM|EM|ZM|YM|DM|CM|MM|ÂµM|NM|PM|FM|A
        M|ZM|YM|MOL)","",maiuscula))!= tamInicial){
      resposta = ""
    }else if
        (nchar(gsub("[/](L|DAL|HL|KL|ML|GL|TL|PL|EL|ZL|YL|DL|CL|LL|ÂµL|NL|PL|FL|
        AL|ZL|YL|MOL)|[0-
        9](L|DAL|HL|KL|ML|GL|TL|PL|EL|ZL|YL|DL|CL|LL|ÂµL|NL|PL|FL|AL|ZL|YL|MO
        L)","",maiuscula))!= tamInicial){
      resposta = ""
    }else if (nchar(gsub("[/](GML|KPH|CUMM)|[0-9](GML|KPH|CUMM)","",maiuscula))!=
        tamInicial){
      resposta = ""
    }
  }
  return (resposta)
}
```

```
##########################################################################
#                    Fim Removedor de Unidades de Medida
##########################################################################
```

Script para atender atividade: Extração de Termos

```
#############################################################################
#                       Inicio Extrator de Termos
#############################################################################

funcaoGetTermos <- function(textoCorpus,endDicionario) {
  resposta<-""
  sp <- removerPalavrasDoDicionario(textoCorpus, endDicionario)
maxPapers <- length(sp);

  for(i in 1:maxPapers){

    if (length(unlist(sp[i]," "))>0){

      maxPalavras <- length(unlist(sp[i]," "));
palavras <- unlist(sp[[i]]);

      for(p in 1:maxPalavras){

        original <- palavras[p];

        if (length(original)>0){

          palavra <- removeCInvalidos(original);

          gs <- ""
          gs <- gsub("[^\x20-\x7E]","",palavra) #remove caratreres nao ascii
          gs <- gsub("[^A-Za-z0-9+(][[:punct:]]{1}[^A-Za-z0-9+)]", "", gs)
          gs <- gsub("[.]", "", gs) #remove ponto
          gs <- gsub("[^A-Za-z\\-][0-9]{1,10}","",gs) #remove numeros sozinhos
          gs <- gsub("[a-z]","",gs)
          gs <- gsub("\\b[A-Z]{1}\\b","",gs) #remove letras maius sozinhas
          gs <- gsub("US\\$","",gs)
          gs<-removeUnidadesMedida(gs)

          if (length(gs) > 0){
            termo <- ""
            termo <- removeCInvalidos(original);
            termo<-gsub("[.]$", "",termo)
            termo<-gsub("[,]$", "",termo)
            #termo<-gsub("[-]$","",termo)
            termo<-gsub("US\\$","",termo)
            termo<-gsub("[0-9](rd|th)","",termo)
termo<-gsub("and/or","",termo)

#remover caracteres html
            termo<-gsub("&amp;","",termo)
            termo<-gsub("&lt;","",termo)
            termo<-gsub("&gt;","",termo)
            termo<-gsub("[0-9](x|X)[0-9]","",termo) #remove formatacao de matrizes e.g 2x2
```

```
        termo<-gsub("p=0.[0-9]*","",termo) #remove p=0.*
        termo<-gsub("P0.[0-9]*","",termo)
        termo<-gsub("i.e","",termo)

        #remove informa??es de medidas micromolar
        termo<-gsub("0.*[0-9]Mol/[a-zA-Z]|0.*[0-9]mol/[a-zA-Z]|0.*[0-9]MOL/[a-zA-
        Z]","",termo)

        #remove informacoes de unidades de medida
        termo<-removeUnidadesMedida(termo)

        #outros tratamentos
        termo <- gsub("[^\x20-\x7E]","",termo) #remove caratreres nao ascii
        termo <- gsub("[^A-Za-z0-9+(][[:punct:]]{1}[^A-Za-z0-9+)]", "", termo)
        termo <- funcaoRemovePontuacao(termo)
        termo <- removeSubstantivosVerbos(termo);
termo <- gsub("[[:punct:]]end", "",termo)
        termo <- gsub("[[:punct:]]and", "",termo)
        termo<-gsub("*HOUR|hour|Hour|hours|Hours|HOURS*","",termo)

apoio <- gsub("[[:punct:]]|[0-9]","",termo); #remove numeros e pontuaÃ§Ã£o

        if ((termo != "") &&
            (nchar(apoio) == nchar(original)) &&
            (apoio!="") &&
            (nchar(apoio)>=3)){
          resposta<-paste(resposta,termo,sep=" ")
        }else if (nchar(gsub("[A-Z]|[0-9]","",termo)) == 0 && nchar(original)>=3 &&
        nchar(apoio)>=3){ #apenas letras maiusculas, Ã© candidato a proteina
          resposta<-paste(resposta,original,sep=" ")
        }else if (nchar(gsub("[a-z]|[-]|[0-9]","",termo))==0 && nchar(original)>=3 &&
        nchar(apoio)>=3 && nchar(gsub("[-]","",termo))<nchar(termo)){#exemplo,
        argonaute-2
          resposta<-paste(resposta,original,sep=" ")
        }else if (nchar(gsub("[A-Z]|[0-9]","",termo))<=3 && nchar(original)>=3 &&
        nchar(apoio)>=3){ # exemplo (ADRBK1)
          resposta<-paste(resposta,original,sep=" ")
        }

       }#fim if gs!= ""
      }#fim lenght(palavra)
    }#fim for max palavras
}#fim if length
  }#fim for maxPapers

return(resposta)
}

##########################################################################
#                    Fim Extrator de proteinas
##########################################################################
```

```
#############################################################################
#                       Inicio Execucao
#############################################################################

executarExtracao <- function (corpus,fetch){

#corpus10041 <- corpus10041[1:14]

  arrayIdCorpus <- array(dim=c(length(corpus),3))
  len <- length(corpus)
  pmid<-PMID(fetch)
  for(i in 1:len){
    if (length(corpus[[i]]) >0){
print(i)
      apoio <- funcaoGetTermos(corpus[[i]],endDicionario)
      arrayIdCorpus[i,][1]<- pmid[i] #pmid
      arrayIdCorpus[i,][2]<- corpus[[i]] #corpus
      arrayIdCorpus[i,][3]<- apoio #termos
      print(apoio)
    }
  }
  return(arrayIdCorpus)
}
```

Script para atender atividade: Salvar relacionamento entre artigos e termos

```
library(RISmed)
source("scriptDatabase.r")
source("scriptTM.r")

#############################################################################
#
#  Inicio armazenar no banco os artigos que retornaram termos
#
#############################################################################

armazenarArtigos <- function(idprocesso, arquivoRData, fetch, arrayIdCorpus){

  #load(arquivoRData)
  con <-conectarPostgres();
  print(idprocesso)
colunaTitulos <- ArticleTitle(fetch)
  len <- length(arrayIdCorpus)/3

  for(i in 1:len){
    if (arrayIdCorpus[i,][3] != ""){
      pmid<-arrayIdCorpus[i,][1] #pmid
      corpus<-arrayIdCorpus[i,][2] #corpus
nomeartigo<-colunaTitulos[i];
      dados <- c(pmid,corpus,nomeartigo,idprocesso)
      print(pmid)
      print(idprocesso)
      salvarArtigo(con,dados)
    }
  }
  dbDisconnect(con);

}

#############################################################################
#
#  Fim armazenar no banco os artigos que retornaram termos
#
#############################################################################

#############################################################################
#
#  Inicio Gerar matrix de termos e seus artigos
#
#############################################################################

relacionaTermoArtigo <- function(arquivoResultados, corpus, arrayIdCorpus) {

  #load(arquivoResultados)
```

```
 len <- length(corpus)
 termos<-" "
 for(i in 1:len){
  termos<-paste(termos, arrayIdCorpus[i,][3], sep=" ")
 }
 termosbkp<-termos

 tabelaTermos <- table(strsplit(termos, " "))
 colunaTermos <- arrayIdCorpus[,3]

 lentable <- length(tabelaTermos)
 arrayArtigoTermo <- array(dim=c(lentable,3))

 inicio <- 1

 for(i in inicio:lentable){
  print (i)
  termoOriginal <- names(tabelaTermos[i]);
  termoSemPontuacaoExtremidade <- funcaoRemovePontuacao(termoOriginal);
  #Tratar Parenteses Chaves Conchetes
  termo <- funcaoTratarPaChCol(termoOriginal);
  if (termo != ""){

    #procura pelo termo na coluna de termos
    #e assim saber qual artigo possui qual termo

    indices <- tryCatch({

      retorno <- ""
      retorno <- grep(termo,colunaTermos);

}, warning = function(war) {

      print(paste("TM_WARNING:  ",war))
      return(retorno)

    }, error = function(err) {

      # warning handler picks up where error was generated
      print(paste("TM_ERROR:  ",err))
      return(nchar(NULL))

}, finally = {

      print(termo)

    })

    idsArtigo<-""

    if (nchar(indices) > 0 ){ #se retornou indices de artigos com esse termo
for (j in 1:length(indices)){
```

```
idsArtigo <- paste(idsArtigo, arrayIdCorpus[indices[j],][1],sep=" ")
        }
        #salva na matrix de artigo x termo tendo como id o pmid
        arrayArtigoTermo[i,][1]<- idsArtigo
        arrayArtigoTermo[i,][2]<- termoSemPontuacaoExtremidade
      }
    }
  }
  return (arrayArtigoTermo)
}

#############################################################################
#
#  Fim Gerar matriz de termos e seus artigos
#
#############################################################################
```

Script para atender atividade: Análise Exploratória

```
#####################################################################
# ANALISE EXPLORATORIA - GRAFICOS
# DENGUE
#####################################################################

##"chagas[ti]   4
##"dengue[ti]          5
##"malaria[ti]  7

load("resultadoPasso2_Processo5.RData")
library(Hmisc)
library(tm)
library(RISmed)


#- Quantos artigos por ano: fora BD   grafico1.png
fetchDengue<-retorno[[3]]
yearDengue<-Year(fetchDengue)
tableYear<-table(yearDengue)

png("graficoArtigosAnoDengue.png")
barplot(tableYear,lwd=2,col=rgb(r=0.0,g=0.4,b=0.7),xlab="Ano",ylab="No. de
        Artigos",las=1,space= 0.5)  #col=22 rosa
title(main="Dengue: Artigos por Ano",sub="fonte: PubMed")
dev.off()

#- Quantos artigos por pai?s: fora BD
Country.data.Dengue<-Country(fetchDengue)
Country.data.Dengue<-lapply(Country.data.Dengue,FUN=tolower)
Country.data.Dengue<-lapply(Country.data.Dengue,FUN=capitalize)
tableCountry.Dengue<-table(unlist(Country.data.Dengue))

tableCountry.maisfreq.Dengue<-sort(tableCountry.Dengue,decreasing=T)
tableCountry.maisfreq.Dengue<-tableCountry.maisfreq.Dengue[1:5]

tableCountryLabels <-
        round(tableCountry.maisfreq.Dengue[][1:5]/sum(tableCountry.Dengue) * 100, 1)
tableCountryLabels <- paste(tableCountryLabels, "%", sep="")

png("graficoArtigosPaisDengue.png")
colors <- c("orange","blue","pink","yellow","green")
pie(tableCountry.maisfreq.Dengue,main="Dengue: Artigos publicados por país",sub="fonte:
        PubMed",
col=colors,label=tableCountryLabels)
legend(1.5, 0.5, names(tableCountry.maisfreq.Dengue[1:5]), cex=0.6,
fill=colors)
dev.off()

#- Quantos artigos foram retornados por doenca (retornando termos ou não): BD
```

```
quantidadeArtigos<-length(PMID(fetchDengue))
#7524

######################################################################
# ANALISE EXPLORATORIA - GRAFICOS
# MALARIA
######################################################################

##"chagas[ti]   4
##"dengue[ti]    5
##"malaria[ti]  7

load("resultadoPasso2_Processo7.RData")

library(Hmisc)
library(tm)
library(RISmed)

#- Quantos artigos por ano: fora BD   grafico1.png
fetchMalaria<-retorno[[3]]
yearMalaria<-Year(fetchMalaria)
#yearMalaria<-yearMalaria[yearMalaria<2013]
tableYearMalaria<-table(yearMalaria)

png("graficoArtigoAnoMalaria.png")
barplot(tableYearMalaria,lwd=2,col=rgb(r=0.0,g=0.4,b=0.7),xlab="Ano",ylab="No. de
        Artigos",las=1,space= 0.5)  #col=22 rosa
title(main="Malaria: Artigos por Ano",sub="fonte: PubMed")
dev.off()

#- Quantos artigos por paÃ-s: fora BD
Country.data.Malaria<-Country(fetchMalaria)
Country.data.Malaria<-lapply(Country.data.Malaria,FUN=tolower)
Country.data.Malaria<-lapply(Country.data.Malaria,FUN=capitalize)
tableCountry.Malaria<-table(unlist(Country.data.Malaria))

tableCountry.maisfreq.Malaria<-sort(tableCountry.Malaria,decreasing=T)
tableCountry.maisfreq.Malaria<-tableCountry.maisfreq.Malaria[1:5]

tableCountryLabels <-
        round(tableCountry.maisfreq.Malaria[][1:5]/sum(tableCountry.Malaria) * 100, 1)
tableCountryLabels <- paste(tableCountryLabels, "%", sep="")

png("graficoArtigosPaisMalaria.png")
colors <- c("orange","blue","pink","yellow","green")
pie(tableCountry.maisfreq.Malaria,main="malária: Artigos publicados por país",sub="fonte:
        PubMed",
col=colors,label=tableCountryLabels)
legend(1.5, 0.5, names(tableCountry.maisfreq.Malaria[1:5]), cex=0.6,
fill=colors)
dev.off()
```

```
#- Quantos artigos foram retornados por doenca (retornando termos ou não): BD
quantidadeArtigos<-length(PMID(fetchMalaria))
#30298


######################################################################
# ANALISE EXPLORATORIA - GRAFICOS
# CHAGAS
######################################################################

##"chagas[ti]   4
##"dengue[ti]    5
##"malaria[ti]  7

load("resultadoPasso2_Processo4.RData")

library(Hmisc)
library(tm)
library(RISmed)

#- Quantos artigos por ano: fora BD   grafico1.png
fetchChagas<-retorno[[3]]
yearChagas<-Year(fetchChagas)
#yearChagas<-yearChagas[yearChagas<2013]
tableYearChagas<-table(yearChagas)

png("graficoArtigoAnoChagas.png")
barplot(tableYearChagas,lwd=2,col=rgb(r=0.0,g=0.4,b=0.7),xlab="Ano",ylab="No. de
        Artigos",las=1,space= 0.5)  #col=22 rosatitle(main="Chagas: Artigos por
        Ano",sub="fonte: PubMed")
title(main="Doença de Chagas: Artigos por Ano",sub="fonte: PubMed")
dev.off()

#- Quantos artigos por paÃ-s: fora BD
Country.data.Chagas<-Country(fetchChagas)
Country.data.Chagas<-lapply(Country.data.Chagas,FUN=tolower)
Country.data.Chagas<-lapply(Country.data.Chagas,FUN=capitalize)
tableCountry.Chagas<-table(unlist(Country.data.Chagas))

tableCountry.maisfreq.Chagas<-sort(tableCountry.Chagas,decreasing=T)
tableCountry.maisfreq.Chagas<-tableCountry.maisfreq.Chagas[1:5]

tableCountryLabels <- round(tableCountry.maisfreq.Chagas[][1:5]/sum(tableCountry.Chagas)
        * 100, 1)
tableCountryLabels <- paste(tableCountryLabels, "%", sep="")

png("graficoArtigosPaisChagas.png")
colors <- c("orange","blue","pink","yellow","green")
pie(tableCountry.maisfreq.Chagas,main="Chagas: Artigos publicados por país",sub="fonte:
        PubMed",
col=colors,label=tableCountryLabels)
legend(1.5, 0.5, names(tableCountry.maisfreq.Chagas[1:5]), cex=0.6,
```

```
fill=colors)
dev.off()

#- Quantos artigos foram retornados por doenca (retornando termos ou não): BD
quantidadeArtigos<-length(PMID(fetchChagas))
#4762


#########################################################################
# ANALISE EXPLORATORIA - TODAS DOENCAS
#########################################################################

1 #- Quantos artigos retornaram termos: BD

select count(a.idartigo), p.chavebusca
from tm.artigo a, tm.processo p
where a.idprocesso = p.idprocesso
group by p.chavebusca

2619;"chagas"
5104;"dengue"
7168;"malaria"

2 #- Quantos artigos retornaram termos vÃ¡lidos: BD

select count(a.idartigo), p.chavebusca
from tm.artigo a, tm.processo p
where a.idprocesso = p.idprocesso and
a.idartigo in
(select c.idartigo
 from tm.artigo_termo c, tm.termo_proteina d
 where c.idtermo = d.idtermo)
group by p.chavebusca

2588;"chagas"
4481;"dengue"
6215;"malaria"

3 #- Quantos termos foram retornados por doenÃ§a: BD

select count(distinct a.idtermo), p.chavebusca
from tm.artigo_termo a, tm.artigo b, tm.processo p
where a.idartigo = b.idartigo and
p.idprocesso = b.idprocesso
group by p.chavebusca

2753;"chagas"
5077;"dengue"
7600;"malaria"

4 #- Quais os termos mais frequentes: BD
```

```
source("scriptTM.r")
source("scriptDatabase.r")

rs<-buscarTermosFrequencia(0);

listaIdentificador<-unlist(rs$idtermo)
listaValor<-unlist(rs$descricao)
listafrequencia<-unlist(rs$total)
listas<-removerItensEspurios(listaIdentificador, listaValor, listafrequencia)
print(length(listas[[1]]))
print(length(listas[[2]]))

termosMaisFrequentes<-c(listas[1],listas[2],listas[3])

write.table(x=termosMaisFrequentes,file="termosMaisFrequentes.xls",qmethod =
        "double",sep=" ")
```

5 #- Quais os termos mais frequentes por doenca: BD

```
source("scriptTM.r")
source("scriptDatabase.r")

rs<-buscarTermosFrequentesPorDoenca();

listaIdentificador<-unlist(rs$chavebusca)
listaValor<-unlist(rs$descricao)
listafrequencia<-unlist(rs$total)
listas<-removerItensEspurios(listaIdentificador, listaValor, listafrequencia)
print(length(listas[[1]]))
print(length(listas[[2]]))

termosMaisFrequentesPorDoenca<-c(listas[2], listas[1],listas[3])

write.table(x=termosMaisFrequentesPorDoenca,file="termosMaisFrequentesPorDoenca.xls",
        qmethod = "double",sep=" ")
```

6 #- Quais termos retornaram ligantes: BD

```
source("scriptTM.r")
source("scriptDatabase.r")

rs<-buscarTermosItensPDB();

listaIdentificador<-unlist(rs$idtermo)
listaValor<-unlist(rs$descricao)
listafrequencia<-unlist(rs$total)
listas<-removerItensEspurios(listaIdentificador, listaValor, listafrequencia)
print(length(listas[[1]]))
print(length(listas[[2]]))

TermosNoPDB<-c(listas[1], listas[2], listas[3])
```

```
write.table(x=TermosNoPDB,file="TermosNoPDB.xls",qmethod = "double",sep=" ")

7 #- Quais termos retornaram farmacos: BD

source("scriptTM.r")
source("scriptDatabase.r")

rs<-buscarTermosComLigante();

listaIdentificador<-unlist(rs$idtermo)
listaValor<-unlist(rs$descricao)
listafrequencia<-unlist(rs$total)
listas<-removerItensEspurios(listaIdentificador, listaValor, listafrequencia)
print(length(listas[[1]]))
print(length(listas[[2]]))

TermosComLigante<-c(listas[1], listas[2], listas[3])

write.table(x=TermosComLigante,file="TermosComLigante.xls",qmethod = "double",sep="
          ")
```

Script para atender atividade: Análise exploratória – Clusterização (dendogramas)

```
criaDendograma<-function(dtm){
  # cria ma matrix de termos
  dados.dtm <- dtm

  # ver a matrix de termos
  dados.dtm

  # ver quais sao os termos mais frequentes
  findFreqTerms(dados.dtm, lowfreq=10)

  # A fun?ao abaixo "simplifica" a matriz
  # altere o parametro sparse , quanto maior mais termos!
  # o interesante e ficar com algo tipo 20 a 30 termos

  dados.dtm2 <- removeSparseTerms(dados.dtm, sparse=0.98)

  # converte a matriz em data.frame, com os termos sendo
  # as colunas e os twitts as linhas
  dados.df <- as.data.frame(inspect(dados.dtm2))

  # verificar o tamanho dos dados
  nrow(dados.df)
  ncol(dados.df)

  # padroniza a escala dos dados

  dados.df.scale <- scale(dados.df)

  # cria uma matriz de distancia euclidiana entre os termos
d <- dist(dados.df.scale, method = "euclidean")

# faz um cluster hierarquico metodo ward
fit <- hclust(d, method="ward")

plot(fit) # plota o dendograma

  grupos <- cutree(fit, k=3) # simplifica a arvore p 3 grupos ,

  # plota o dendograma mostrando os grupos
rect.hclust(fit, k=2, border="red")
}


##########################################
# DENGUE
##########################################
source("scriptDatabase.r")
```

```
rs<-artigosTermosDengue()
save.image("dtmDengue1806.RData")

## script gerado na reunião do dia 13 de maio
#names(rs)
#rs$termos
#rs$idartigo
resp <- cbind(rs$idartigo,rs$termos)
#dim(resp)
#resp[,2]
#strsplit(resp[,2],",")
r2 <- strsplit(resp[,2],",")
names(r2) <- rs$idartigo
library(tm)
Corpus(VectorSource(rs$termos))
teste <- Corpus(VectorSource(rs$termos))
#teste[[1]]
#Corpus(VectorSource(sapply(teste,strsplit,",")))
corpo <- Corpus(VectorSource(sapply(teste,strsplit,",")))
TermDocumentMatrix(corpo)
dtm <- TermDocumentMatrix(corpo)
criaDendograma(dtm)
save.image("dtmDengue1806.RData")

##############################################
# CHAGAS
##############################################
source("scriptDatabase.r")

rs<-artigosTermosChagas()
save.image("dtmChagas1806.RData")

## script gerado na reunião do dia 13 de maio
#names(rs)
#rs$termos
#rs$idartigo
resp <- cbind(rs$idartigo,rs$termos)
#dim(resp)
#resp[,2]
#strsplit(resp[,2],",")
r2 <- strsplit(resp[,2],",")
names(r2) <- rs$idartigo
library(tm)
Corpus(VectorSource(rs$termos))
teste <- Corpus(VectorSource(rs$termos))
#teste[[1]]
#Corpus(VectorSource(sapply(teste,strsplit,",")))
corpo <- Corpus(VectorSource(sapply(teste,strsplit,",")))
TermDocumentMatrix(corpo)
dtm <- TermDocumentMatrix(corpo)
criaDendograma(dtm)
save.image("dtmChagas1806.RData")
```

```
##########################################
# MALARIA
##########################################
source("scriptDatabase.r")

rs<-artigosTermosMalaria()
save.image("dtmMalaria1806.RData")

## script gerado na reunião do dia 13 de maio
names(rs)
rs$termos
#rs$idartigo
resp <- cbind(rs$idartigo,rs$termos)
#dim(resp)
#resp[,2]
#strsplit(resp[,2],",")
r2 <- strsplit(resp[,2],",")
names(r2) <- rs$idartigo
library(tm)
Corpus(VectorSource(rs$termos))
teste <- Corpus(VectorSource(rs$termos))
#teste[[1]]
#Corpus(VectorSource(sapply(teste,strsplit,",")))
corpo <- Corpus(VectorSource(sapply(teste,strsplit,",")))
#TermDocumentMatrix(corpo)
dtm <- TermDocumentMatrix(corpo)
criaDendograma(dtm)
save.image("dtmMalaria1806.RData")
```

Script para atender atividade: Validação: Busca Uniprot

```
################################################################################
#  PASSO 4: BUSCANDO AS INFORMACOES QUE VALIDAM O TERMO
#          COMO VÁLIDO PARA BUSCA DE INFORMACOES SOBRE
#          PROTEINAS NO UNIPROT
################################################################################

source("scriptDatabase.r")
source("scriptTM.r")

queryUniprotEspecifico<- function (
  term,                              # query term
  baseUrl=getOption("serviceUrl.uniprot")   # URL of the PubMed service
){

  library("XML")
  options("serviceUrl.uniprot" = "http://www.uniprot.org/")

if (is.null (baseUrl)) {
    stop ("Acertar a URL do servico do Uniprot!!!")
  }
  #http://www.uniprot.org/uniprot/?query=name%3A%223-
          glucanase%22+OR+gene%3A%223-glucanase%22&sort=score
query<- paste (baseUrl,
                "uniprot/?",
                "query=name%3A%22",
                term,
                "%22+OR+gene%3A%22",
                term,"%22&force=yes&format=xml&",
                "limit=50&sort=score",
                sep="")

  print(query);
  # parse resulting XML into a tree to be returned to user
  result.xml = ""
  tryCatch(
    result.xml<- xmlTreeParse(file=query),
    error= return(queryUniprotGeral(term))
  )
  return (result.xml)
}

queryUniprotGeral<- function (
  term,                              # query term
  baseUrl=getOption("serviceUrl.uniprot")   # URL of the PubMed service
){

  library("XML")
  options("serviceUrl.uniprot" = "http://www.uniprot.org/")
```

```
if (is.null (baseUrl)) {
   stop ("Acertar a URL do servico do Uniprot!!!")
}

  query<- paste (baseUrl,
                 "uniprot/?query=",
                 term,
                 "&sort=score&format=xml&",
                 "limit=50&",
                 sep="")



  print(query);
  # parse resulting XML into a tree to be returned to user
  result.xml = ""
  tryCatch(
    result.xml<- xmlTreeParse(file=query),
    finally=return(result.xml)
  )

}

buscaUniprot <- function(arrayArtigoTermo){

  #source("scriptDatabase.r")
  inicio<-1
  lentable <- length(arrayArtigoTermo)/3
  infoUniprot <- ""
  termo <- ""
  con <-conectarPostgres();
  for(i in inicio:lentable){

    if (!is.na(arrayArtigoTermo[i,][2])){
      #print (i)
      dados<-""
      resXML<-""
      print(nchar(infoUniprot))
      #if (nchar(infoUniprot) >0){
      if (!(is.null(infoUniprot)) && (nchar(infoUniprot) >0)){
#print(termo)
        #armazena no banco
        resXML <- XML::toString.XMLNode(infoUniprot$doc[[1]])
        #Pego o nome das proteinas
        #Fim pego nome das proteinas
resXML <- gsub("","""",resXML)
dados <- c(termo,resXML)
      }else{
        dados <- c(termo,"")
      }
      if (termo!=""){
```

```
        idTermo <- buscarIdTermoPorDescricao(con,termo);

        if (idTermo == ""){
          #salva o termo jÃ¡ trazendo o id do termo
          idTermo <- salvarTermo(con,dados)
        }
        print(paste("idtermo===",idTermo,sep=""))

        #salva relacionamento entre termo e artigos
        salvarTermoListaArtigo(con,idTermo,idsArtigo)

        #print(paste("infouniprot==",infoUniprot,sep=""))
        #print("salvei termo lista artigo")

if (!is.null(infoUniprot) && nchar(infoUniprot)>0){
        print("tem dado uniprot")

        parseado <- XML::append.XMLNode(infoUniprot$doc[[1]],to=xmlNode(name=""))
#salva proteinas encontradas no XML
        indice <-1
print(indice)
        totalEntries <- length(parseado[1]$uniprot[])
        #print(totalEntries)

        while (indice < totalEntries){
          #print("indice < totalEntries")

          indiceAccession <- 1
          idUniprot <-
          xmlValue(parseado[1]$uniprot[indice]$entry[indiceAccession]$accession) #accession

          while (!is.null(parseado[1]$uniprot[indice]$entry[indiceAccession]$accession)){
            indiceAccession <- indiceAccession + 1
          }

          #print("cheguei")
          nomeUniprot <-
          xmlValue(parseado[1]$uniprot[indice]$entry[indiceAccession]$name)  #entryName
          indiceAccession <- indiceAccession + 1
          nomeCompleto <-
          xmlValue(parseado[1]$uniprot[indice]$entry[indiceAccession]$protein[1]$recommen
          dedName[1]$fullName) #fullName
          if (nchar(nomeCompleto) <= 2){
            nomeCompleto <-
          xmlValue(parseado[1]$uniprot[indice]$entry[indiceAccession+1]$protein[1]$recomm
          endedName$fullName) #fullName
}

          dadosProteina <- c(nomeUniprot,idUniprot,nomeCompleto);
          idProteina <-salvarProteina(dadosProteina);

          dadosTermoProteina <- c(idProteina,idTermo);
```

```
          salvarTermoProteina(dados=dadosTermoProteina);

          indice <- indice + 1
          #print("somando 1")
         }

       }
       #print("nem entrei")
      }

     idsArtigo <- arrayArtigoTermo[i,][1]
     termo <- arrayArtigoTermo[i,][2]
if (termo != ""){
       infoUniprot <- ""
       tryCatch(
infoUniprot<- queryUniprotEspecifico(gsub("^[[:punct:]]|[[:punct:]]$","",termo)),
finally=next
       )
     }
   }
  }

  if (nchar(infoUniprot) >0){
#armazena no banco
   resXML <- XML::toString.XMLNode(infoUniprot$doc[[1]])
resXML <- gsub("'","''",resXML)
dados <- c(termo,resXML)
  }else{
   dados <- c(termo,"")
  }

  if (termo!=""){

   idTermo <- buscarIdTermoPorDescricao(con,termo);
   #print(idTermo)
   if (idTermo == ""){
    #salva o termo jÃ¡ trazendo o id do termo
    idTermo <- salvarTermo(con,dados)
   }
   #salva relacionamento entre termo e artigos
   salvarTermoListaArtigo(con,idTermo,idsArtigo)

   if (nchar(infoUniprot) >0){
    parseado <- XML::parseXMLAndAdd(XML::toString.XMLNode(infoUniprot$doc[[1]]))
    #salva proteinas encontradas no XML
indice <-1
while (parseado[1]$uniprot[indice]$entry[1] == FALSE){
       #accession:
       accession <- parseado[1]$uniprot[indice]$entry[1][[1]][[1]]
#name:
       proteinName <- parseado[1]$uniprot[indice]$entry[2][[1]][[1]]
       dadosProteina <- c(proteinName,accession);
```

```
      idProteina <-salvarProteina(dadosProteina);

      dadosTermoProteina <- c(idProteina,idTermo);
      salvarTermoProteina(dados=dadosTermoProteina);

      indice <- indice + 1;
    }
   }
  }

  dbDisconnect(con);

}
```

Script para atender atividade: Validação: Busca PDB

```
#################################################################
#  PASSO 5: BUSCANDO AS INFORMACOES NO PDB
#################################################################

#load("resultados_AntesBuscaUniprot.RData")

library(XML)
library(RISmed)
library(rJava)

source("scriptDatabase.r")
source("scriptTM.r")

preparaXMLPorAccession <- function(accession){
#Prepara o XML utilizado para a busca por accession no PDB
  xml <-
          "<orgPdbQuery><queryType>org.pdb.query.simple.UpAccessionIdQuery</queryTyp
          e>"
xml <- paste(xml, "<description>Simple query for a list of UniProtKB
          Accession</description>")
  xml <- paste(xml, "<accessionIdList>",accession,"</accessionIdList>")
xml <- paste(xml, "</orgPdbQuery>")
}

preparaXMLPorTermo <- function(termo){
  #Prepara o XML utilizado para buscar dados textuais no PDB
  xml <-
          "<orgPdbQuery><queryType>org.pdb.query.simple.AdvancedKeywordQuery</query
          Type><description>"
xml <- paste(xml, "Text Search for: ",termo,"</description><keywords>",termo);
  xml <- paste(xml,"</keywords></orgPdbQuery>");
  return (xml)
}

doPOST<-function(url,encodedXML){
#decodifica o xml utilizado para busca de informacoes no PDB
conn <- url$openConnection();
  conn$setDoOutput(TRUE);

  wr <- new(J("java.io.OutputStreamWriter"),conn$getOutputStream());
  wr$write(encodedXML);
  wr$flush();
  res <- conn$getInputStream();

  return (res)

}

postQuery <- function(xml){
```

```
  #realiza a busca por accession via servico Rest Webservice do PDB
  serviceLocation <- "http://www.rcsb.org/pdb/rest/search/?sortfield=Resolution";
  url <- new(J("java.net.URL"), serviceLocation)
encoder <- J("java.net.URLEncoder")
  encodedXML <- encoder$encode(xml,"UTF-8")
res <- doPOST(url,encodedXML)
  is <- new(J("java.io.InputStreamReader"),res)
  bf <- new(J("java.io.BufferedReader"),is)
  return (bf)
}

queryPDBFile <- function(pdb){

  query<-paste("http://www.rcsb.org/pdb/rest/describePDB?structureId=",pdb,sep="")

  print(query);

  result.html = ""
  tryCatch(
    result.html<- htmlTreeParse(file=query),
    finally=return(result.html)
  )

}

getLigantesPDB <- function(idpdb,idproteina){
  serviceLocation<-
          paste("http://www.rcsb.org/pdb/rest/ligandInfo?structureId=",idpdb,sep="")
  result.html<-htmlTreeParse(file=serviceLocation)

  result.names<-unlist(result.html)

chemicalids<-NULL
  ligantes<-NULL
  formulas<-NULL
for (i in 1:length(result.names)){
    #print(i)
    noAtual <- unlist(strsplit(names(result.names[i]),"\\.")," ")
    result.grep <- grep("chemicalid",noAtual)
    if ((length(result.grep)>0) && (result.grep == length(noAtual))){ #Trata-se do chemicalid
      chemicalids<-c(chemicalids,result.names[[i]])
    }
    result.grep <- grep("chemicalname",noAtual)
    length.grep <- length(noAtual)-3
    result.grep.value <- grep("value",noAtual)
    length.grep.value <- length(noAtual)

    if ((length(result.grep)>0) && (length(result.grep.value)>0)){
      if ((result.grep == length.grep) &&
(result.grep.value == length.grep.value)){ #Trata-se do chemicalname
        ligantes<-c(ligantes,result.names[[i]])
}
```

```
    }

    result.grep <- grep("formula",noAtual)
    length.grep <- length(noAtual)-3
    result.grep.value <- grep("value",noAtual)
    length.grep.value <- length(noAtual)

    if ((length(result.grep)>0) && (length(result.grep.value)>0)){
      if ((result.grep == length.grep) &&
(result.grep.value == length.grep.value)){ #Trata-se da formula
        formulas<-c(formulas,result.names[[i]])
}
    }

  }
  if (!is.null(ligantes)){
    for (i in 1:length(ligantes)){
hetname<-ligantes[i]
      formula<-formulas[i]
      chemicalid<-chemicalids[i]
      idligante<-salvarLigante(hetname,formula,chemicalid)
      salvarPdbLigante(idpdb,idligante,idproteina)
    }
  }
  return(c(ligantes,formulas))

}

buscaPDB <- function(idprocesso,inicio,lenrs){
  #
  #
  #Ver http://deposit.pdb.org/cc_dict_tut.html#Anchor-Recor-25695
#
  source("scriptDatabase.r")

  .jinit() #nao esquecer!!!

#busca informacoes no banco
  rs <- buscarTodasProteinasPorIdProcesso(idprocesso);
#print(rs)
  if (inicio == 0){
    inicio <- 1
  }
  if (lenrs == 0){
    lenrs <- length(rs$idproteina)
}
  bf <- NULL

  # enquanto tiverem termos no banco
for(i in inicio:lenrs){
    print(i)
```

```
   if (!is.null(bf)){
     line<-"";
     pdbIds<-"";
     line = bf$readLine();

     while (!is.null(line)) {
       pdbId <- line;
pdbFile <- NULL

       #Sys.sleep(15) #aguardar 15 segundos para o PDB nÃ£o reclamar

pdbFile <- queryPDBFile(pdbId)

       if (pdbFile != ""){
         dadosPdbId<-unlist(pdbFile); #$children$html[1]$body[1]$pdbdescription[1]$pdb
descricao <- dadosPdbId[[10]] #title;
         dados <- c(pdbId, descricao, idproteina);
         salvarPDB(dados);

         ligantes<- getLigantesPDB(pdbId,idproteina);

pdbIds<-paste(pdbId, line, sep=" ")
       }

       line = bf$readLine();#pdbid retornado

     }
     #print(paste("ids:",pdbIds,sep=""))
     bf$close();
   }

   idproteina <- rs[[1]][i]
   accession <- rs[[2]][i]

   print(accession)

   bf <- NULL
   xml<-preparaXMLPorAccession(accession)
#Sys.sleep(15) #aguardar 15 segundos para o PDB nÃ£o reclamar
tryCatch(
     bf <- postQuery(xml)
   )
 }
 #print("fim")
 if (!is.null(bf)){
   line<-"";
   pdbIds<-"";
   line = bf$readLine();

   while (!is.null(line)) {
     pdbId <- line;
```

```
      #print(pdbId)

      pdbFile <- queryPDBFile(pdbId)

      if (pdbFile != ""){
        dadosPdbId<-unlist(pdbFile);#$children$html[1]$body[1]$pdbdescription[1]$pdb);
descricao <- dadosPdbId[[10]] #title;
        dados <- c(pdbId, descricao, idproteina);
        salvarPDB(dados);

        ligantes<- getLigantesPDB(pdbId,idproteina);
#print(ligantes);

        pdbIds<-paste(pdbId, line, sep=" ")
      }

      #pdbFile$close();
      line = bf$readLine();#pdbid retornado

    }
    print(paste("ids:",pdbIds,sep=""))
}
  bf$close();

}
```

Script para atender atividade: Busca Ligantes

```
################################################################################
#  PASSO 6: BUSCANDO AS INFORMACOES NO DRUG BANK
#          http://www.drugbank.ca/search/advanced
################################################################################

source("scriptDatabase.r")
source("scriptTM.r")

queryDrugBank<- function (
  term,                          # query term
  type,                          # tipo da busca
  baseUrl=getOption("serviceUrl.drugbank") # URL of the DrugBank service
){

  library("XML")
  options("serviceUrl.drugbank" = "http://www.drugbank.ca/")

  if (is.null (baseUrl)) {
    stop ("Acertar a URL do servico do Drugbank!!!")
  }
  #http://www.drugbank.ca/search?query=pdb_id:1LS6

          #http://www.drugbank.ca/search?query=mixture_ingredient%3Adextromethorphan+A
          ND+mixture_ingredient%3Adoxylamine

  options(useFancyQuotes = FALSE)

  if (type == "hetid"){
    #term<-"PCA"
    query<- paste (baseUrl,
                   "search?query=het_id:",
                   term,
                   sep="")
  }else{
    #term<-"PYROGLUTAMIC ACID"
    query<- paste(baseUrl,"search?query=name:",
              dQuote(term),sep="");
  }
  print(query);

  result.html = ""
  tryCatch(
    result.html<- htmlTreeParse(file=query),
    finally=return(result.html)
  )
}

getDrugBankFile<- function (link){
```

```
  print(link);

  result.html = ""
  tryCatch(
    result.html<- htmlTreeParse(file=link),
    finally=return(result.html)
  )


}


getDadoDrugBank <- function(idprocesso){

rs<-buscarLigantesPorProcesso(idprocesso);

  lenrs <- length(rs$idligante);
  inicio <-1
  arrayResposta<-""

  #lenrs<-8
  for(i in inicio:lenrs){
print(i)

    drugbank.item.accession <-""
    drugbank.item.het<-""
    drugbank.item.name<-""

    hetid <- rs[[1]][i]
    hetname <- rs[[2]][i]
    idligante <- rs[[3]][i]

    drug.result<-queryDrugBank(hetid,"hetid");
    drug.result<-unlist(drug.result)

    result.grep <- grep("/drugs/",drug.result)

    linkdrugbank <- ""

    if (length(result.grep)>0){ #Se encontrou o item
      linkdrugbank<-paste("http://www.drugbank.ca",drug.result[[result.grep[[1]]]],sep="")
    }else{
      drug.result<-queryDrugBank(hetname,"name");
      drug.result<-unlist(drug.result)

      result.grep <- grep("/drugs/",drug.result)

if (length(result.grep)>0){ #Se encontrou o item
        linkdrugbank<-paste("http://www.drugbank.ca",drug.result[[result.grep[[1]]]],sep="")
}
    }
```

```
   #print(paste("LINK DRUG BANK:",linkdrugbank,sep=""))

   if (linkdrugbank != ""){

     #linkdrugbank<-"http://www.drugbank.ca/drugs/DB03564"

     fileDrugBank<-getDrugBankFile(linkdrugbank)
     fileDrugBank.result<-unlist(fileDrugBank)

#Pegar o nome do item no DrugBank
result.grep <- grep("Name",fileDrugBank.result)

if (length(result.grep)>0){ #Se encontrou o item
drugbank.item.name<-fileDrugBank.result[[result.grep[[2]]+4]]
      #print(paste("ITEM-NAME:",drugbank.item.name,sep=""))
}

    #Pegar o valor do id do hetname no DrugBank
result.grep <- grep("CODE",fileDrugBank.result)

if (length(result.grep)>0){ #Se encontrou o item

      #Calcula o tamanho do link para extrair o numero do item

texto.link <-fileDrugBank.result[[result.grep[[1]]]]
      start.link <-nchar(texto.link)-2

      drugbank.item.het<-substr(texto.link,start=start.link,stop=nchar(texto.link))
      #print(paste("ITEM-HET-CHEMID:",drugbank.item.het,sep=""))

}

    #Pegar o accession do item no DrugBank
result.grep <- grep("Accession",fileDrugBank.result)

if (length(result.grep)>0){ #Se encontrou o item
drugbank.item.accession<-fileDrugBank.result[[result.grep[[1]]+4]]
      #print(paste("ITEM-ACCESSION:",drugbank.item.accession,sep=""))
}

    #Salvar os dados: dbid, nome, idligante, hetid
    salvarDrugBank(c(drugbank.item.accession,drugbank.item.name,drugbank.item.het))
salvarLiganteDrugBank(drugbank.item.accession,idligante)

   }
   linkdrugbank <- ""
  }

  return(arrayResposta)

}
```

```
#/drugs/DB04417
#> g<-grep("ul",dadosDrug)
#[1] 256 258 271 276 281 286 291 296 301 306 307
#> dadosDrug[[256]]
#[1] "/drugs/DB04417"

#resultado<-getDadoDrugBank(15)
```

Script para atender a busca de estruturas similares

```
#############################################################################
#  PASSO 7: BUSCANDO INFORMACOES DO PDB.ORG:
#          Sequence Clustering and Redundancy Reduction Results
#############################################################################
source("scriptDatabase.r")
source("scriptTM.r")
library("XML")

#Calculate pairwise sequence or structure alignments.
#query<-
        "http://www.rcsb.org/pdb/workbench/workbench.do?action=pw_smithwaterman&nam
        e1=1ls6&name2=2d06"
#for(i in 1:2000){
#  print(i)
#  result.html<- htmlTreeParse(file=query)
#  print(result.html)
#}


#cutoff 40%
#query<-
        "http://www.rcsb.org/pdb/explore/sequenceCluster.do?structureId=4HHB&entity=1&s
        eqid=40"

getSequenciasSimilaridadesEstrutura<-function(idprocesso,cutoff,ini,lenrs){
  #cutoff: valor de corte que definimos como melhores sendo 40% e 50%
  #idprocesso<-15

  rs<-buscarPDBPorProcesso(idprocesso);

  if (ini == 0){
ini <- 1
  }
  if (lenrs == 0){
    lenrs <- length(rs$idpdb)
  }

  #print(paste("length(rs$idpdb):",length(rs$idpdb),sep=""))

similares<-NULL
  taxsimilares<-NULL
  for(i in ini:lenrs){

    idpdb <- rs[[1]][i]
    print(paste("valor do i: ",i, " para ",idpdb,sep=""))
#idpdb<-"2BSS"

    query<-paste("http://www.rcsb.org/pdb/explore/sequenceCluster.do?structureId=",
              idpdb,"&entity=1&seqid=",cutoff,sep="")
```

```
   result.html<- htmlTreeParse(file=query)
   result<-unlist(result.html)
   result.indice<-grep("checkbox",result)
   result.indice.taxonomy<-grep("External Link to NCBI Taxonomy Entry",result)

inicio<-2 #o primeiro Ã© o check box do select all
lenresult<-length(result.indice)

   print(paste("lenresult: ",lenresult,sep=""))

   if (lenresult>0){
     for(indice in inicio:lenresult){
       #print(indice)
       idpdbSimilar<-result[[result.indice[[indice]]+3]]
       #print(idpdbSimilar)

achou<-FALSE
       indicetaxon<-31
       taxonomia<-""
       while((achou!=TRUE) && (indicetaxon < 39)){
         taxonomia<-result[[result.indice[[indice]]+indicetaxon]] #para responder quais sÃ£o
           homo sapiens
         tam<-nchar(taxonomia)
         if (tam != nchar(gsub("External Link to NCBI Taxonomy Entry - ","",taxonomia))){
           achou<-TRUE
           taxonomia<-gsub("External Link to NCBI Taxonomy Entry - ","",taxonomia)
         }else{
           indicetaxon<-indicetaxon+1
           taxonomia<-""
         }
       }

       #print(idpdbSimilar)
       similares<-paste(similares,idpdbSimilar,sep=" ")
       taxsimilares<-paste(taxsimilares,taxonomia,sep=",")

     }
   }
   salvarSimilaresTaxon(idpdb,similares,taxsimilares)

   similares<-NULL
taxsimilares<-NULL
  }
}

getArrayPDBSimilaridades<-function(ini,lenrs,min,max,limite){

  rs<-buscarPDBsSimilares(min,max,limite);

  if (ini == 0){
    ini <- 1
```

```
  }
  if (lenrs == 0){
lenrs <- length(rs$idpdb)
  }

  retorno<-NULL
  indiceRetorno<-1
for(i in ini:lenrs){

idproteina<-rs[[1]][i]
    idpdb <- rs[[2]][i]
    descricao <- rs[[3]][i]
    similares <- rs[[4]][i]
    taxsimilares <- rs[[5]][i]

    print(paste("valor do i: ",i, " para ",idpdb,sep=""))

listaSimilares<-strsplit(similares," ")
    listaTaxSimilares<-strsplit(taxsimilares,",")

    for(indice in 2:length(listaSimilares[[1]])){

#indice=1 é sempre vazio

      similar<-listaSimilares[[1]][indice]
      tax<-listaTaxSimilares[[1]][indice]

      linha<-NULL

      linha<- paste(idpdb, similar, tax,sep=",")

      indiceRetorno<-indiceRetorno+1

      retorno<-paste(retorno,linha,sep=";")

    }
  }
  return(retorno)
}
```

Script para atender a busca de dados das famílias das proteínas

```
################################################################################
#  PASSO 8: BUSCANDO INFORMACOES SOBRE A FAMILIA DA PROTEINA
################################################################################

source("scriptDatabase.r")
source("scriptTM.r")

queryUniprotFamily<- function (idproteina)
{

  library("XML")

  query<- paste ("http://pfam.sanger.ac.uk/protein/",idproteina,sep="")

  print(query);
  # parse resulting XML into a tree to be returned to user
  result.xml = ""
  tryCatch(
    result.html<- htmlTreeParse(file=query),
    finally=return(result.html)
  )
}

getFamilias<-function(idprocesso,inicio,lenrs){

#rs <- buscarTodasProteinasPorIdProcesso(idprocesso);

  #especificos:
  rs<-buscarProteinasFrequentesUniprot();

#print(rs)
  if (inicio == 0){
    inicio <- 1
  }
  if (lenrs == 0){
    lenrs <- length(rs$idproteina)
}
  infoUniprot <- ""

  # enquanto tiverem termos no banco
for(i in inicio:lenrs){

    print(i)

    if (!is.null(infoUniprot) && nchar(infoUniprot)>0){
      print("tem dado uniprot")

      dados<-unlist(infoUniprot$children)
      dados.indice<-grep("http://pfam.sanger.ac.uk/family/",dados)
```

```
if (length(dados.indice)>0){ #Se encontrou o item
        familia<-dados[[dados.indice[1]]]

        print(paste("FAMILIA:",familia,sep=""))

        salvarFamiliaProteina(idproteina,familia)
      }

    }

    idproteina <- rs[[1]][i]
accession <- rs[[2]][i]

    infoUniprot <- ""
    tryCatch(
      infoUniprot<- queryUniprotFamily(accession)
    )
  }

  if (!is.null(infoUniprot) && nchar(infoUniprot)>0){
    print("tem dado uniprot")

    dados<-unlist(infoUniprot$children)
    dados.indice<-grep("http://pfam.sanger.ac.uk/family/",dados)

if (length(dados.indice)>0){ #Se encontrou o item
      familia<-dados[[dados.indice[1]]]

      print(paste("FAMILIA:",familia,sep=""))

      salvarFamiliaProteina(idproteina,familia)
    }

  }

}
```

Script para atender a busca de organismos que acusam presença da proteína

```
################################################################################
#  PASSO 9: BUSCANDO INFORMACOES SOBRE ORGANISMO EM QUE A
          PROTEINA REGISTRADA FOI ENCONTRADA
################################################################################

source("scriptDatabase.r")
source("scriptTM.r")

queryUniprotOrganism<- function (idproteina)
{

  library("XML")

  query<- paste ("http://www.uniprot.org/uniprot/",idproteina,".xml",
                 sep="")

  print(query);
  # parse resulting XML into a tree to be returned to user
  result.xml = ""
  tryCatch(
    result.xml<- xmlTreeParse(file=query),
finally=return(result.xml)
  )
}

getOrganismoProteinas<-function(inicio,lenrs,idprocesso){

  rs <- buscarTodasProteinasPorIdProcesso(idprocesso);
#print(rs)
  if (inicio == 0){
    inicio <- 1
  }
  if (lenrs == 0){
    lenrs <- length(rs$idproteina)
}
  infoUniprot <- ""

  # enquanto tiverem termos no banco
for(i in inicio:lenrs){

    print(i)

    if (!is.null(infoUniprot) && nchar(infoUniprot)>0){
print("tem dado uniprot")

      dados<-unlist(infoUniprot$doc[[1]])
      dados.indice<-grep("organism",dados)
      dados.indiceLineage<-grep("lineage",dados)
```

```
      if (length(dados.indice)>0){ #Se encontrou o item
        organismo<-""
        for(idado in dados.indice:dados.indiceLineage){
          familia<-dados[[idado]]
          if (familia=="scientific"){
            organismo<-dados[[idado+2]]
          }
        }

        #familia<-dados[[dados.indice[1]+9]]

        print(paste("ORGANISM:",organismo,sep=""))

        salvarOrganismoProteina(idproteina,organismo)
}

    }

    idproteina <- rs[[1]][i]
    accession <- rs[[2]][i]

    infoUniprot <- ""
    tryCatch(
      infoUniprot<- queryUniprotOrganism(accession)
    )
  }

  if (!is.null(infoUniprot) && nchar(infoUniprot)>0){
print("tem dado uniprot")

    dados.indice<-grep("organism",dados)
    dados.indiceLineage<-grep("lineage",dados)

    if (length(dados.indice)>0){ #Se encontrou o item
      organismo<-""
      for(idado in dados.indice:dados.indiceLineage){
        familia<-dados[[idado]]
        if (familia=="scientific"){
          organismo<-dados[[idado+2]]
        }
      }

      #familia<-dados[[dados.indice[1]+9]]

      print(paste("ORGANISM:",organismo,sep=""))

      salvarOrganismoProteina(idproteina,organismo)
    }

  }

}
```

Script para atender atividade: Geração de Grafos

```
#############################################################################
#       CONSTRUCAO DOS GRAFOS
#############################################################################

geraGrafo<-function(colunaFrom,colunaTo,isString,nomearquivo){

  ########################################
  #
  #ATENCAO: NOME DO VERTICE NÃƒO PODE CONTER ESPACOS
  #
  ########################################

  library(igraph)

  #edge-aresta ou linha que liga os nÃ³s
#vertex-vertice(no)

  g <- graph.empty()
  vertices<-names(c(table(colunaFrom),table(colunaTo)))

  g <- add.vertices(g, length(vertices),
                    name=vertices,label=vertices)

  #funciona para testar g<-add.edges(g, as.character(traits[,1]) )

  # Extract first names from the full names
  if (isString == TRUE){
    names <- sapply(V(g)$name, "[",1)
  }else{
    names <- sapply(strsplit(V(g)$name, " "), "[",1)
  }

  ids <- 1:length(names)
  names(ids) <- names

  # Create the edges
  from <- as.character(colunaFrom)
  to <- as.character(colunaTo) #as.character(unlist(rs$idtermo))
  edges <- t(matrix(c(ids[from], ids[to]),nc=2))

  g <- add.edges(g, edges,desc=vertices)

  E(g)$color <- "black"
  V(g)$color <-  "white" #rainbow(length(vertices))
  tkplot(g, layout=layout.kamada.kawai, edge.color=E(g)$color,vertex.label=vertices)

  if (nomearquivo != ""){
    write.graph(g, paste(nomearquivo,".gml",sep=""), "gml")
  }
```

```
  return(g)

}


#### TERMO X PDB -  Artigos que se relacionam pelo farmaco recuperado (FEITO para
          denv e atp)


source("scriptDatabase.r")

#rs<-dadosGrafoTermoxPDBPorTermos("'ATP'")


rs<-dadosGrafoTermoxPDBPorFrequenciaMinMax(11,10)#(8588,6720)
rs<-dadosGrafoTermoxPDBPorTermos("'DENV'")
colunaFrom<-as.character(unlist(rs$nometermo))
colunaTo<-as.character(unlist(rs$nomepdb))

g<-geraGrafo(colunaFrom, colunaTo, TRUE,"dadosGrafoTermoxPDBPorTermosDENV")


#### FARMACO x ARTIGO
source("scriptDatabase.r")
rs<-dadosGrafoFarmacoArtigo(10,40)
length(rs$nomefarmaco)

colunaFrom<-as.character(unlist(rs$nomefarmaco))
colunaTo<-as.character(unlist(rs$pmid))

g<-geraGrafo(colunaFrom, colunaTo, TRUE,"dadosGrafoFarmacoArtigo")

#### TERMO FREQUENTE X DOENCA

source("scriptDatabase.r")
rs<-dadosGrafoTermoFrequenteDoenca(5,7)
length(rs$termo)

colunaFrom<-as.character(unlist(rs$termo))
colunaTo<-as.character(unlist(rs$doenca))

g<-geraGrafo(colunaFrom, colunaTo, FALSE,"TermoxDoenca")

#### PROTEINA FREQUENTE X DOENCA

source("scriptDatabase.r")
rs<-dadosGrafoProteinaFrequenteDoenca(6)
length(rs$proteina)

colunaFrom<-as.character(unlist(rs$proteina))
colunaTo<-as.character(unlist(rs$doenca))
```

```
g<-geraGrafo(colunaFrom, colunaTo, FALSE,"ProteinaxDoenca")

#### PROTEINA FREQUENTE X DOENCA

source("scriptDatabase.r")
rs<-dadosGrafoLiganteDoenca(5,5,7,80)
length(rs$hetname)

colunaFrom<-as.character(unlist(rs$chemicalid))
colunaTo<-as.character(unlist(rs$doenca))

g<-geraGrafo(colunaFrom, colunaTo, FALSE,"LigantexDoenca")

#### FARMACO X DOENCA

source("scriptDatabase.r")
rs<-dadosGrafoFarmacoDoenca(5,7,100)
length(rs$doenca)

colunaFrom<-as.character(unlist(rs$dbid))
colunaTo<-as.character(unlist(rs$doenca))

g<-geraGrafo(colunaFrom, colunaTo, TRUE,"FarmacoxDoenca")
write.graph(g, "Ligantexdoenca.gml","gml")

#### PROTEINA X FAMILIA

source("scriptDatabase.r")
rs<-dadosGrafoProteinaFamilia(9,0)
length(rs$proteina)

colunaFrom<-as.character(unlist(rs$proteina))
colunaTo<-as.character(unlist(rs$familia))

g<-geraGrafo(colunaFrom, colunaTo, TRUE,"proteinaxfamilia")
write.graph(g, "proteinaxfamilia.gml","gml")

#### PROTEINA X ORGANISMO (HOMO SAPIENS)

source("scriptDatabase.r")
rs<-dadosGrafoProteinaOrganismo(16,0)
length(rs$proteina)

colunaFrom<-as.character(unlist(rs$proteina))
colunaTo<-as.character(unlist(rs$organismo))

g<-geraGrafo(colunaFrom, colunaTo, TRUE,"proteinaxorganismo")

#### PROTEINA X SIMILARES X HOMO SAPIENS

source("scriptDatabase.r")
source("scriptTM7.r")
```

```
rs<-buscarPDBsSimilares(9,9,0)  #amostra de 50 linhas dos PDBS similares com frequencia
        de termo = 9
length(rs$idpdb)

r<-getArrayPDBSimilaridades(0,0,9,9,0)
l<-strsplit(r,";")
write.table(x=l[[1]][2:length(l[[1]])],file="retornoTodos990.txt",row.names =
        FALSE,col.names=FALSE)

similares <- read.csv("retornoTodos990.txt",sep=",",quote="")
similares[1,][1]
pdbs.all<-similares[][1]
pdbs.simi<-similares[][2]
pdbs.taxsimi<-similares[][3]

sort(table(pdbs.all))

grep.res<-grep("2PX5",unlist(similares[][1]))  #anterior: 1C2P  #4H3Q

pdbs<-similares[grep.res,][1]
codsimilares<-similares[grep.res,][2]
taxsimilares<-similares[grep.res,][3]

length(table(pdbs))
length(table(codsimilares))
length(table(taxsimilares))

colunaFrom<-as.character(unlist(taxsimilares))
colunaTo<-as.character(unlist(pdbs))

g<-geraGrafo(colunaFrom, colunaTo, TRUE,"proteinaxTaxonomySimilares")
write.graph(g, "proteinaxTaxonomySimilares.gml","gml")

source("scriptDatabase.r")
rs<-dadosGrafoFarmacoArtigo(40)
length(rs$nomefarmaco)

colunaFrom<-as.character(unlist(rs$nomefarmaco))
colunaTo<-as.character(unlist(rs$pmid))

g<-geraGrafo(colunaFrom, colunaTo, TRUE,"dadosGrafoFarmacoArtigo.gml")

# #### ARTIGO X ARTIGO (ARTIGOS RELACIONADOS PELO TERMO) - Artigos que
        se relacionam pelo termo mais frequente
#
# source("scriptDatabase.r")
# # tem termo = RNA rs<-dadosGrafoArtigosLigadosPeloTermo(4,29871)
# rs<-dadosGrafoArtigosLigadosPeloTermo(4,33671)
# length(rs$url1)
#
```

```
# #colunaFrom<-
        c(as.character(unlist(rs$artigo1)),as.character(unlist(names(table(rs$artigo2)))))
# colunaFrom<-as.character(unlist(rs$artigo2))
# colunaTo<-as.character(unlist(rs$termo1))
#
# g<-geraGrafo(colunaFrom, colunaTo, FALSE,"artigoxtermoxartigo")
# write.graph(g, "artigoxtermoxartigo.gml","gml")
```

Script para geração da nuvem de termos frequentes

```
library(tm)
library(wordcloud)
library(RColorBrewer)

#############################################################################
# WORDCLOUD CHAGAS
#############################################################################

source("scriptDatabase.r")
rs<-termosChagas()
termos<-rs$termos
xkcd.corpus <- Corpus(VectorSource(termos))
#
xkcd.corpus <- tm_map(xkcd.corpus, removePunctuation)
xkcd.corpus <- tm_map(xkcd.corpus, tolower)
xkcd.corpus <- tm_map(xkcd.corpus, function(x) removeWords(x, stopwords("english")))
tdm <- TermDocumentMatrix(xkcd.corpus)
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
pal <- brewer.pal(9, "BuGn")
pal <- pal[-(1:2)]
png("chagas.png", width=1280,height=800)
wordcloud(d$word,d$freq, scale=c(8,.3),min.freq=2,max.words=100, random.order=T,
        rot.per=.15, colors=pal, vfont=c("sans serif","plain"))
dev.off()

#############################################################################
# WORDCLOUD MALARIA
#############################################################################
source("scriptDatabase.r")
rs<-termosMalaria()
termos<-rs$termos
xkcd.corpus <- Corpus(VectorSource(termos))
#
xkcd.corpus <- tm_map(xkcd.corpus, removePunctuation)
xkcd.corpus <- tm_map(xkcd.corpus, tolower)
xkcd.corpus <- tm_map(xkcd.corpus, function(x) removeWords(x, stopwords("english")))
tdm <- TermDocumentMatrix(xkcd.corpus)
tdmbkp<-tdm
tdm <- removeSparseTerms(tdmbkp, sparse=0.9999)
#tdm <- removeSparseTerms(tdmbkp, sparse=0.9995)
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
pal <- brewer.pal(8, "Dark2")
pal <- pal[-(1:2)]
png("malaria.png", width=1280,height=800)
```

```
wordcloud(d$word,d$freq, scale=c(8,.3),min.freq=2,max.words=100, random.order=T,
        rot.per=.15, colors=pal, vfont=c("sans serif","plain"))
dev.off()


################################################################################
# WORDCLOUD DENGUE
################################################################################
source("scriptDatabase.r")
rs<-termosDengue()
termos<-rs$termos
xkcd.corpus <- Corpus(VectorSource(termos))
#
xkcd.corpus <- tm_map(xkcd.corpus, removePunctuation)
xkcd.corpus <- tm_map(xkcd.corpus, tolower)
xkcd.corpus <- tm_map(xkcd.corpus, function(x) removeWords(x, stopwords("english")))
tdm <- TermDocumentMatrix(xkcd.corpus)
tdmbkp<-tdm
tdm <- removeSparseTerms(tdmbkp, sparse=0.9999)
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
pal <- brewer.pal(8, "Dark2")
pal <- pal[-(1:2)]
png("dengue.png", width=1280,height=800)
wordcloud(d$word,d$freq, scale=c(8,.3),min.freq=2,max.words=50, random.order=T,
        rot.per=.15, colors=pal, vfont=c("sans serif","plain"))
dev.off()
```