AUTARQUIA ASSOCIADA À UNIVERSIDADE DE SÃO PAULO

Métodos e softwares para análise da produção científica e detecção de frentes emergentes de pesquisa

José Sergio Bleckmann Reis Junior

Dissertação apresentada como parte dos requisitos para obtenção do Grau de Mestre em Ciências na Área de Tecnologia Nuclear - Reatores

Orientador:
Prof. Dr. Antonio Carlos de Oliveira Barroso

São Paulo
2015

INSTITUTO DE PESQUISAS ENERGÉTICAS E NUCLEARES
Autarquia associada à Universidade de São Paulo

Métodos e softwares para análise da produção científica e detecção de frentes emergentes de pesquisa

José Sergio Bleckmann Reis Junior

Dissertação apresentada como parte dos requisitos para obtenção do Grau de Mestre em Ciências na Área de Tecnologia Nuclear - Reatores

Orientador:
Prof. Dr. Antonio Carlos de Oliveira Barroso

Versão Corrigida
Versão Original disponível no IPEN

São Paulo
2015

Este trabalho é dedicado à minha irmãzinha, Suzana, e a todos os professores que lecionaram, lecionam ou lecionarão no Ensino Fundamental II, no Ensino Fundamental I e na Educação Infantil, em especial aos meus colegas professores do Colégio Catamarã, e de modo muitíssimo especial aos meus pais, Prof.ª Cecília e Prof. Sergio: é preciso coragem e amor quase infinitos para, por uma vida inteira, se dedicar a formar as crianças, estes pequenos seres infinitamente mais complexos e mais apaixonantes que o próprio problema P versus NP*.*

AGRADECIMENTOS

Agradeço a todos os professores do IME-USP e do IPEN–CNEN/SP, e também a todos os professores de cujas aulas eu tive a honra de participar na pós-graduação, pelo conhecimento amplo que me proporcionaram e pelo gosto pela ciência que me tansmitiram, em especial aos Profs. Drs. Kengo Imakuma e Mario de Menezes, do IPEN–CNEN/SP, e Prof. Dr. Diógenes Bido, da Universidade Presbiteriana Mackenzie, grandes amigo meus, exemplos de dedicação aos alunos e à Ciência.

Agradeço aos meus colegas da pós-graduação, por tudo o que sofremos e rimos juntos, em especial a Rita Ricciardi, Felipe Kiiper, Marcelo Seixas, Egon Hannes, Anisio Monteiro, Beth Acar, Fernando Moreira e Luis Gustavo Almeida: o companheirismo de todos deixou a jornada, às vezes muitíssimo pesada, sempre mais leve.

Agradeço ao Prof. Dr. Antônio Carlos de Oliveira Barroso, cuja inteligência, dedicação e caráter me transmitiram grande amor pela docência e pela pesquisa científica.

Agradeço à minha irmãzinha, Suzana, que, apesar de não se entusiasmar com Ciência da Computação, aplica tecnologia de ponta na transformação de vidas através de sua linda profissão de terapeuta oupacional e, deste modo, me entusiasma cada vez mais a fazer Ciência da Computação com paixão e excelência.

Agradeço muitíssimo aos meus pais, Cecília e Sergio, que me educaram na fé católica, me deram minha irmãzinha tão amada, não pouparam sacrifícios por mim e por minha irmã, me colocaram na faculdade, não permitiram que eu desistisse, me animaram a cursar o mestrado, torcem pelo meu doutorado, e são os mais felizes com o meu mestrado em Tecnologia Nuclear.

“The hardest thing is to go to sleep at night, when there are so many urgent things needing to be done. A huge gap exists between what we know is possible with today’s machines and what we have so far been able to finish.” (Donald Knuth)

MÉTODOS E SOFTWARES PARA ANÁLISE DA PRODUÇÃO CIENTÍFICA E DETECÇÃO DE FRENTES EMERGENTES DE PESQUISA

José Sergio Bleckmann Reis Junior

RESUMO

O progresso de projetos anteriores salientou a necessidade de tratar o problema dos softwares para detecção, a partir de bases de dados de publicações científicas, de tendências emergentes de pesquisa e desenvolvimento. Evidenciou-se a carência de aplicações computacionais eficientes dedicadas a este propósito, que são artigos de grande utilidade para um melhor planejamento de programas de pesquisa e desenvolvimento em instituições. Foi realizada, então, uma revisão dos softwares atualmente disponíveis, para poder-se delinear claramente a oportunidade de desenvolver novas ferramentas. Como resultado, implementou-se um aplicativo chamado *Citesnake*, projetado especialmente para auxiliar a detecção e o estudo de tendências emergentes a partir da análise de redes de vários tipos, extraídas das bases de dados científicas. Através desta ferramenta computacional robusta e eficaz, foram conduzidas análises de frentes emergentes de pesquisa e desenvolvimento na área de Sistemas Geradores de Energia Nuclear de Geração IV, de forma que se pudesse evidenciar, dentre os tipos de reatores selecionados como os mais promissores pelo GIF - *Generation IV International Forum*, aqueles que mais se desenvolveram nos últimos dez anos e que se apresentam, atualmente, como os mais capazes de cumprir as promessas realizadas sobre os seus conceitos inovadores.

MÉTODOS E SOFTWARES PARA ANÁLISE DA PRODUÇÃO CIENTÍFICA E DETECÇÃO DE FRENTES EMERGENTES DE PESQUISA

José Sergio Bleckmann Reis Junior

ABSTRACT

The progress of previous projects pointed out the need to face some problems of software for detecting emerging research and development trends from databases of scientific publications. It became evident the lack of efficient computing applications dedicated to this purpose that are artifacts of great usefulness to better planning research and development programs in institutions. A review of the currently available software was performed, in order to clearly delineate the opportunity to develop new tools. As a result, a software called *Citesnake* was implemented, designed particularly to help the detection and study of emerging trends from the analysis of networks of several types extracted from the scientific databases. Using this robust and effective computational tool, analyzes of emerging research and development trends were performed in the field of Generation IV Nuclear Power Generation Systems, in such a way to point out, among the most promising reactor types selected by the GIF - Generation IV International Forum, those that have better evolved over the past ten years and seem to be currently the most capable of fulfilling the promises made on their innovative concepts.

SUMÁRIO

I SOFTWARES PARA ANÁLISE DA PRODUÇÃO CIENTÍFICA 11

1 INTRODUÇÃO 12
1.1 Motivação e oportunidade da pesquisa 12
1.2 Justificativa da proposta de estudo 14

2 OBJETIVOS 17
2.1 Revisão da literatura pertinente 17
2.2 Problemas propostos e objetivos 21

3 SOFTWARES 25
3.1 Fundamentos teóricos para o *Citesnake* 25
3.2 Exploração dos softwares existentes e opção pelo Citesnake . 28
3.3 Conclusão do Citesnake beta 4 31
3.4 O Citesnake e os scripts de Leydesdorff 34

II MÉTODOS PARA DETECÇÃO DE FRENTES EMERGENTES DE PESQUISA 35

4 MÉTODOS 36
4.1 Análise quantitativa das publicações 37
4.2 Análise das redes de publicações 38

5 RESULTADOS 43
5.1 Instituição x autor 43
5.2 Autor x artigo 43
5.3 Autor x autor 44
5.4 Artigo x palavra-chave 46

5.5 Artigos x referências 46
5.6 Fragmentação das redes monomodais 46

6 ANÁLISE E DISCUSSÃO DOS RESULTADOS 48

7 CONCLUSÕES E TRABALHOS FUTUROS 50

APÊNDICES 53

APÊNDICE A – CÓDIGO DO CITESNAKE 54

APÊNDICE B – CÓDIGO DO MÓDULO CSNAKE_WOS_MOD . 92

APÊNDICE C – CÓDIGO DO MÓDULO CSNAKE_SCP_MOD . . 95

APÊNDICE D – CÓDIGO DO MÓDULO CSNAKE_GSC_MOD . . 98

APÊNDICE E – CÓDIGO DO MÓDULO CSNAKE_PMD_MOD . . 100

LISTA DE ILUSTRAÇÕES

FIGURA 1 – Fluxograma do trabalho realizado para a solução dos problemas levantados 24

FIGURA 2 – Número de registros por ano, segundo Web of Science e Scopus, referenciando no título, no resumo ou nas palavras-chave, cada um dos seis tipos de reatores estudados, e *não necessariamente* mencionando algum dos dez marcos técnico-científicos da evolução dos conceitos de reatores inovadores 39

FIGURA 3 – Número de registros por ano, segundo Web of Science e Scopus, referenciando no título, no resumo ou nas palavras-chave, cada um dos seis tipos de reatores estudados, e *necessariamente* mencionando algum dos dez marcos técnico-científicos da evolução dos conceitos de reatores inovadores 40

FIGURA 4 – Número de registros por ano, segundo Web of Science, Scopus e Google Scholar, referenciando no título, no resumo ou nas palavras-chave, cada um dos seis tipos de reatores estudados, e *não necessariamente* mencionando algum dos dez marcos técnico-científicos da evolução dos conceitos de reatores inovadores 41

FIGURA 5 – Autores recorrentemente bem classificados quanto às medidas usuais para redes de colaboração 45

LISTA DE TABELAS

TABELA 1 – As dez instituições com maior número de autores ativos na área analisada 43
TABELA 2 – Os dez autores com maior número de publicações no campo analisado 44
TABELA 3 – As dez palavras-chave mais frequentes nas publicações analisadas 46
TABELA 4 – As dez referências mais citadas pelos registros analisados . . . 47

Parte I

SOFTWARES PARA ANÁLISE DA PRODUÇÃO CIENTÍFICA

1 INTRODUÇÃO

1.1 Motivação e oportunidade da pesquisa

O avanço dos sistemas de computação vem proporcionando, em especial na última década, a implementação e disponibilização de bancos de dados contendo vastíssima quantidade de publicações científicas. Servidores de alto desempenho e algorítmos avançados permitem hoje à comunidade científica de todo o planeta consultar, em pouquíssimo tempo, uma quantidade de artigos e informações científicas impensável há cerca de vinte anos atrás. De fato, a produção técnico-científica mundial vem crescendo a taxas elevadas: estima-se que, em apenas dezenove anos - de 1991 a 2008 – o número de publicações científicas por ano em todo o mundo cresceu cerca de 171% [1, 2]. A utilização, por parte dos pesquisadores, de tamanha quantidade de informação seria impossível sem o emprego de recursos computacionais extremamente potentes. Imensos bancos de dados científicos com recursos de busca otimizada passaram então a ser implementados em diversos países, sob a iniciativa tanto pública quanto privada, abarcando diversas áreas do conhecimento.

Paralelamente, nestes últimos trinta anos, os Métodos Formais para Análise de Redes Sociais ganharam a atenção de muitos cientistas, em especial devido ao seu “atraente foco nas relações entre as entidades sociais, e nos padrões e implicações destas relações” [4]. A perspectiva das redes tornou-se ferramenta poderosa para o tratamento de inúmeras questões latentes das ciências sociais e comportamentais. Mas, de fato, tal ferramenta não ficou restrita apenas a esse campo do conhecimento: os métodos da Análise de Redes Sociais encontraram aplicações da Psicologia à Epidemiologia, da Sociologia à Bibliometria, da Topologia de Redes Computacionais à Gestão Estratégica de Empresas. E, finalmente, com o advento dos referidos bancos de dados de literatura científica, tornou-se possível aplicar tais métodos ao estudo das características, estrutura e dinâmica da produção técnico-científica mundial. A enorme quantidade de dados disponível é agora passível de um tratamento sistemático, que conduza à aquisição de informações para tratar questões de singular importância como, por exemplo, a dinâmica mundial da produção e

transferência do conhecimento.

Entretanto, existe o problema da implementação de softwares para análise de redes sociais que, por si só, tem se apresentado como uma área que não se pode classificar como “esgotada”. Aplicativos muito diversos tem sido continuamente desenvolvidos e aprimorados, com suporte aos mais diversos formatos de dados, com capacidade de calcular as mais diversas medidas e com potentes geradores de gráficos dos mais diversos tipos – alguns deles foram analisados e comparados por Huisman e van Duijn [24]. Não obstante, não há como que uma “solução definitiva” para analisar redes; os pesquisadores dedicados à área, quando não iniciam a implementação de novas ferramentas para levar a cabo seus objetivos mais singulares, têm trabalhado com *pipelines* integradores dos softwares de interesse, como é mesmo sugerido nos manuais e tutoriais dos softwares para análise de redes sociais disponíveis. O panorama é um pouco menos animador quando nos restringimos aos aplicativos capazes de analisar redes a partir dos dados exportados por bancos de literatura científica. Eles existem em número muito menor que os softwares devotados a redes mais gerais, e costumam trazer um cardápio de funcionalidades muito limitadas, o que frustra a expectativa dos usuários. Desta forma, permanece a necessidade de “malabarismos” com os dados, que recorrentemente precisam ser “jogados” de um programa a outro e com o agravante que os dados provenientes de bases de literatura geram redes multimodais, o que complica sua conversão. Esse problema vem atraindo a atenção de alguns grupos de pesquisa dedicados à chamada *Ciência da Informação Científica*.

Vimos, pois, nessa lacuna, a oportunidade de um trabalho de importância singular para a área: a análise dos softwares atualmente disponíveis para a condução de pesquisas sobre redes – de diversos tipos – da literatura científica, análise esta que proporcionasse delinear claramente tanto as possibilidades e impossibilidades de tais aplicativos como a oportunidade de desenvolvimento de novas ferramentas e bibliotecas de funções. Uma vez constatada tal oportunidade, pretendeu-se implementar o núcleo de um software que possibilitasse aos interessados levar a termo tais pesquisas sem a necessidade de excessivos “malabarismos computacionais” ou, em outras palavras, aplicativos que definitivamente proporcionem ao usuário preocupar-se apenas com a sua área de estudo, e não com a programação de computadores. De fato, a existência e a funcionalidade de uma tal ferramenta são

decisivas para o desenvolvimento de pesquisas que almejem, através da aplicação das técnicas de *Análise da Informação Científica*, aferir informações precisas sobre o desenvolvimento da Ciência e da Tecnologia, suas fronteiras e as redes de colaboração que envolvem, informações estas que se mostram, atualmente, bastante relevantes para muitos setores da sociedade. A título de exemplo, nas aplicações de prospecção tecnológica sobre como um determinado processo, tecnologia ou produto deve – ou pode – evoluir nos próximos anos, há a necessidade de grande agilidade na extração de significado das pesquisas bibliométricas, um problema que ainda não fora resolvido satisfatoriamente com as ferramentas disponíveis.

1.2 Justificativa da proposta de estudo

Egghe e Rosseu [6] já afirmavam que as técnicas da *Análise de Citações*, que constitui apenas um dos muitos tópicos estudados na *Ciência da Informação Científica*, possuem uma gama de aplicações de singular importância, entre elas: "1) a avaliação qualitativa e quantitativa de cientistas, publicações e instituições de pesquisas; 2) a modelagem do desenvolvimento histórico da Ciência e da Tecnologia; 3) a busca e recuperação da informação". Hoje, tais aplicações se tornaram assuntos de interesse estratégico não só para cientistas, como para universidades, instituições de fomento, empresas de base tecnológica e qualquer organização ligada a pesquisa e desenvolvimento. Sobre esse tema, Garfield e Welljams-Dorof [27] apontam que

> "autores de políticas públicas, gestores de pesquisa corporativa e administradores de universidades precisam de indicadores sobre Ciência e Tecnologia válidos e confiáveis para uma variedade de propósitos: por exemplo, para medir a eficácia das despesas de pesquisa, identificar as áreas fortes e de excelência, estabelecer prioridades para o planejamento estratégico, monitorar seu desempenho em relação aos seus pares e concorrentes, e identificar especialidades emergentes e novas tecnologias para desenvolvimento acelerado" [27].

Susan Cullen [28], consultora da Thomson Reuters, declara ser "imperativo que as organizações saibam quem as está citando e por quê. Tal análise pode proporcionar descobertas sobre a atividade do concorrente, identificar potenciais parceiros e ajudar a proteger portfolios de propriedade intelectual".

Constatações deste gênero adquirem especial significado para o setor nuclear brasileiro. Se desde a década de 80 os investimentos destinados ao setor

observaram severo declínio, e a política energética brasileira consolidou sua preferência pela geração hídrica, o aparente “renascimento da energia nuclear” que os especialistas observavam despontar na década passada sofreu duro golpe com as ocorrências na usina de Fukushima, Japão, provocadas pelo fortíssimo terremoto de 11 de março de 2011. Sendo o setor energético, em especial a fonte nuclear, de relevância estratégica inquestionável para qualquer nação, consideramos que as técnicas da *Ciência da Informação Científica* são de grande proveito às instituições do setor nuclear brasileiro, que se vêem obrigadas a gerir e desenvolver, com recursos bastante restritos, uma tecnologia extremamente complexa e multidisciplinar. De fato, a pesquisa na área nuclear no Brasil se desdobrou em uma rica variedade de aplicações tanto energéticas como não energéticas, abrindo um leque de muitas frentes emergentes de investigação científica, algumas das quais vêm se consolidando como áreas de pesquisa significativamente importantes para a sociedade: podemos citar, como exemplo, a fabricação de radioisótopos para radioterapia e braquiterapia e o estudo de materiais especiais para radioproteção, cada uma com sua comunidade de pesquisadores dedicados, com jornais especializados e com uma estrutura de desenvolvimento particular. Um cenário como este deixa patente a oportunidade do emprego dos métodos da *Ciência da Informação Científica* como ferramenta para a excelência da gestão da pesquisa e desenvolvimento do setor nuclear nacional, o que salienta a necessidade de um “arsenal” estável, confiável e potente de aplicativos computacionais que conduzam gestores e cientistas, através de análises da informação científica, a tal fim.

Seguindo essa direção, o Departamento de Energia dos EUA, no início da década de 2000, liderou uma iniciativa denominada “Generation IV International Forum” (GIF). Dela participaram vários países, inclusive o Brasil, que esteve bastante ativo na fase inicial, até 2002 e, posteriormente, se afastou. O GIF pretendeu, através de um esforço coletivo, descobrir quais novos conceitos de sistemas de geração de energia de base nuclear seriam os mais promissores para um futuro mais sustentável. Neste sentido, foram definidos requisitos de segurança, minimização de rejeitos, eficiência energética, redução dos riscos de proliferação e de segurança física das instalações que deveriam ser atingidos. No âmbito da AIEA, uma iniciativa semelhante surgiu alguns anos depois, mas esta se abriu a uma participação mais ampla de países e instituições. Ela foi denominada INPRO – “International Initiative

on Innovative Reactors and Fuel Cycles". Há, de fato, muitos relatórios e artigos que abordam como marcharam as pesquisas colaborativas originadas pelas iniciativas GIF e INPRO; no entanto, há poucos estudos que abordem, do ponto de vista de resultados concretos, que sistemas evoluíram mais ou, em outras palavras, quais aqueles que, aparentemente, serão os vencedores em termos de possibilidade de implementação. Como um dos objetivos do nosso projeto, pretendemos identificar, através de softwares e métodos desenvolvidos neste trabalho, dentre aqueles sistemas (de reator e ciclo) selecionados como os mais promissores na lista do GIF, os que realmente mais se desenvolveram nos últimos 10 anos. Também procuramos identificar, dentre os mais evoluídos, aqueles que, tendo em vista suas necessidades e a base tecnológica do Brasil, poderiam ser considerados os mais adequados para nossos esforços de pesquisa e desenvolvimento.

Por fim, é importante salientar que o setor nuclear, desde a década de 70 até final do século, adquiriu características extremamente conservadoras, e também a sinalização das potências dominantes na época faziam na prática com que ideias criativas fossem sempre deixadas de lado. Um exemplo emblemático é que o ciclo de combústível "once through" tornou-se padrão, o que no mínimo é um atentado à conservação de recursos naturais. Até mesmo as questões de segurança e não proliferação sempre foram tratadas de maneira não criativa e sob a máxima do "é proibido pensar diferente". Na lógica e ética vigentes neste novo milênio, criatividade e inovação para otimizar objetivos ligados à sustentabilidade fazem parte do credo da comunidade científica, e a área nuclear não poderá ficar de fora. Espera-se um próximo decênio fértil em novas ideias no setor nuclear, semelhante ao que aconteceu nas décadas de 50 e 60, porém com a diferença que os desenvolvimentos deverão acontecer de forma colaborativa, com várias organizações, empresas e países participando. Haverá, portanto, necessidade de estudos prospectivos para orientar os responsáveis pelas políticas de P&D do setor.

2 OBJETIVOS

2.1 Revisão da literatura pertinente

O pioneiro da compilação dos índices de citações, a matéria-prima dos futuros bancos de dados on-line de publicações, foi o cientista americano Eugene Garfield. Em 1960, um ano antes de terminar seu doutorado em Linguística Estrutural, Garfield fundou o Instituto para Informação Científica – ISI, em língua inglesa. Este institituto foi o responsável pela criação do Science Citation Index (SCI) e seus derivados – como o Social Sciences Citation Index (SSCI) e o Arts and Humanities Citation Index (A&HCI) – que por muito tempo consistiram em índices impressos, até que se tornou possível sua digitalização e disponibilização na Internet. Garfield e seu instituto foram os pioneiros da indexação e análise de citações: seus trabalhos constituíram a base da Bibliometria, um conjunto de métodos para a análise quantitativa da literatura científica e tecnológica [3]. O ISI foi adquirido, em 1992, pela Thomson Scientific & Healthcare, e o conteúdo dos seus índices está hoje disponível on-line através da Web of Knowledge [17], que possuí em seu banco de dados os artigos publicados em mais de 14000 períodicos técnico-científicos.

Derek J. de Solla Price foi o primeiro pesquisador a analisar redes sociais construídas através de dados sobre a literatura científica. Em seu artigo intitulado “Network of Scientific Papers”, publicado na revista Science de julho de 1965, o autor analisa não uma rede cujos nós – os atores da rede – são cientistas, mas sim uma rede na qual os nós são artigos científicos publicados; dois nós são, então, ditos relacionados se há, em um deles, uma citação ao outro em nota de rodapé ou na bibliografia. De fato, Price não tinha disponível a enorme quantidade de dados que hoje é acessível através dos bancos científicos on-line: ele utilizou indexes compilados por Garfield e outros pesquisadores no final da década de 50 e início da década de 60.

Entre outras conclusões, de Solla Price observou que as redes de papers que se citam podem ser enquadradas em duas formas gerais: (i) as redes nascidas de frentes emergentes de pesquisa, mais concentradas; o autor observa que os pesquisadores ligados a essas redes, para estarem em compasso com os trabalhos

de seus colegas e pares, deveriam fazer uso de serviços de alerta, disponíveis na época apenas através dos próprios índices de citações; e (ii) as redes de papers que costumam citar todas as obras precedentes do seu campo de conhecimento: são trabalhos que, em certo sentido, tratam toda nova contribuição como parte integrante do conhecimento humano perene e, de tempos em tempos, todo esse conhecimento é sistematizado em livros. O autor encerra seu artigo mencionando a possibilidade de um delineamento da topografia da literatura científica então corrente; ele afirma que, com um estudo deste tipo, provavelmente ser-se-ia capaz de "indicar a sobreposição e a importância relativa de jornais e, de fato, de países, autores ou papers individuais através do lugar que ocuparam no mapa, e pelo grau de centralidade estratégica dentro de uma dada faixa" (de Solla Price, 1965).

Vinte e cinco anos mais tarde, Egghe e Rousseau publicaram o livro Introduction to Informetrics: Quantitative Methods in Library, Documentation and Information Science: é uma obra que almeja reunir os principais conceitos, métodos e resultados dos trabalhos realizados nas duas décadas precedentes no nascituro campo da chamada Informetria – que "trata da mensuração, e portanto da teoria e modelagem matemáticas de todos os aspectos da informação, de seu armazenamento e recuperação" [6]. Esses trabalhos eram, de fato, bastante dispersos uns dos outros. Até então, a maioria dos pesquisadores dedicados a esse novo campo do conhecimento consideravam-no como "sua atividade secundária". A própria nomenclatura da área era alvo de intensa discussão: era comum o uso, além do termo "Informetria", de "Bibliometria" e de "Cientometria". Mas os trabalhos possuiam um objetivo comum: a aplicação da Teoria da Informação à Informação, ou seja, uma meta-informação matemática da literatura técnico-científica. E no livro de Egghe e Rosseau, a Análise de Citações é apontada como ferramenta fundamental para esse campo do conhecimento: logo após os capítulos introdutórios de Métodos Estatísticos e Fundamentos de Pesquisa Operacional, os autores dedicam um capítulo inteiro à teoria da Análise de Citações – desde a compilação de indexes de citações às aplicações às Políticas da Pesquia Científica. Em especial, na terceira seção, é abordada a teoria das Redes de Citações: a contribuição mais notável desta seção é a compilação de Teoremas de Redes de Citações, com demonstrações formais e aplicações da Teoria dos Grafos, e a expressão, em notação matricial, de conceitos ligados aos processos de publicação e citação.

Na década de 90, foram publicados alguns poucos estudos de redes da literatura técnico-científica, constituindo essencialmente dois tipos principais de trabalho: os que analisaram redes de co-autoria, como os artigos de Wouters e Leydesdorff [7], Melin e Persson [8, 9], Ding, Foo e Chowdhury [10] e de Kundra e Kretschmer [11], e os que analisaram redes de co-citação, como os artigos de Persson e Beckmann [12], Usdiken e Pasadeos [13], White e McCain [14], Kleinberg [15] e Chen [16]. Entretanto, essa parece ter sido uma década onde a nova área de pesquisa ainda não houvera se estabelecido plenamente: o número de artigos publicados por ano sobre redes da literatura científica, segundo dados da Web of Science [17], nunca passou de 7. Todos estes trabalhos utilizavam como fonte os bancos de dados do ISI: o Science Citation Index e seus derivados similares. Já no período de 2000 a 2011, observou-se um alvorecer deste campo de pesquisa, quando teve início a multiplicação de trabalhos e de bancos de dados utilizados. Entre artigos envolvendo redes de co-autoria e co-citação, a Web of Science [17] computa 353 publicados nesse período, sendo a média de publicações para seu último quarto de 51 trabalhos por ano, e a média, para o período inteiro, de citações destes trabalhos, de 258 citações por ano.

Os artigos sobre redes da literatura científica publicados nesta última década não se restringiram ao Science Citation Index de Garfield e seus derivados. De fato, novas iniciativas de compilação de bases de dados da literatura técnico científica em todo o mundo puderam ser levadas a termo com a evolução dos sistemas de computação e dos algorítmos de busca, e os cientistas da “Informetria” ou “Ciência da Informação” – ou “Ciência da Ciência”, como alguns pretendem – passaram a ter à disposição uma gama enorme de dados provenientes de outras bases, como Scopus [18], PubMed [19], arXiv [20], Google Scholar [21], CiteSeer [22], que estão entre as mais citadas. Em dezembro de 2011, a Wikipedia [23] disponibilizava uma lista com 129 bancos de dados de publicações técnico-científicas, entre serviços gratuítos e pagos, de diversas empresas e instituições de ensino de mundo. Entretanto, toda essa diversidade, enquanto proporciona enormes possibilidades aos pesquisadores, gera um problema que não se pode qualificar como inesperado: a falta de padronização de exportação de dados entre os bancos. Em primeira análise, parece algo fácil de ser contornado, com alguns “malabarismos” de importação, conversão e exportação entre softwares. Mas a experiência de quem tem se dedicado à

área tem deixado claro que não se pode lidar com massivas quantidades de dados com “improvisos computacionais” sem arcar com uma quantidade de erros bastante alarmante.

Pesquisadores da Universidade de Maryland e da Universidade de Michigan têm trabalhado em um projeto denominado iOpener (Information Organization for PENning Expositions on Research) [25], cujo objetivo é

> “gerar pesquisas prontamente utilizáveis de diferentes domínios e temas científicos, direcionados para diferentes públicos e níveis, por exemplo, especialistas, cientistas de disciplinas afins, educadores, estudantes, gestores públicos e os cidadãos, incluindo as minorias e os grupos sub-representados” [25].

Como parte deste projeto, está em desenvolvimento o ASE (Action Science Explorer), um software que almeja apresentar a literatura científica e seus campos de diferentes formas: listas de artigos, textos completos, visualizações da estrutura de redes de citações, entre outras. Até dezembro de 2011, tal ferramenta estava disponível apenas para os colaboradores dos desenvolvedores, porém sua descrição fazia notar o seguinte: o software é dependente de um conjunto de dados denominado ACL Anthology Network, que reúne, segundo informações dos desenvolvedores, mais de 16000 artigos científicos. Esta característica pode limitar o trabalho do pesquisador. O número de papers adicionados ao banco não é pequeno, mas está muito abaixo da quantidade de dados disponíveis, por exemplo, na Web of Science, que, por exemplo, em apenas uma busca, pelo termo “Information Science”, retorna mais de 80000 artigos.

O software aberto a todo o público que mais se destaca em compilar grandes redes de colaboração científica é, atualmente, o Sci^2 Tool [26], desenvolvido por cientistas da Universidade de Indiana. Essa ferramenta é capaz de importar dados dos bancos científicos mais utilizados, e permite análises temporais, geoespaciais, tópicas, análise de redes e visualizações. Não obstante, a utilização deste aplicativo evidencia a existência de problemas nos métodos responsáveis por consolidar dados provenientes de diferentes bases, quando da necessidade de construir redes através das informações de mais de um banco de dados científico.

2.2 Problemas propostos e objetivos

O estudo da literatura pertinente, bem como o trabalho com os aplicativos utilizados ou indicados pelos principais grupos de pesquisa dedicados à área da *Ciência da Informação Científica*, evidenciaram a existência dos seguintes problemas:

1. Ausência de softwares, tanto gratuitos como proprietários, que sejam capazes, ao mesmo tempo ou, pelo menos, em cadeia eficiente, de:
    a) processar os registros de publicações contidos nos dados de forma eficiente quanto ao tempo e à memória;
    b) comunicar erros de processamento de forma clara, para que o usuário possa corrigir inconsistências dos dados provenientes dos bancos científicos;
    c) consolidar, numa mesma rede, dados exportados por bancos científicos diversos, sem ambiguidade;
    d) calcular medidas de produtividade e medidas características das redes assim que estas forem construídas;
    e) oferecer modularidade e escalabilidade que permitam, com facilidade, a implementação de extensões para lidar com outros formatos de dados e bases científicas;
    f) gerar arquivos de dados íntegros e consistentes, que possam alimentar programas de análise mais avançada;
    g) poder ser utilizados por pesquisadores das mais diversas áreas e formações, gestores públicos, avaliadores de instituições de fomento, administradores de organizações, enfim, pelos mais diversos tipos de usuário, através de uma interface gráfica clara, objetiva e intuitiva, que antes auxilie do que confunda o utilizador,
2. Ausência de metodologia consolidada para a condução de estudos de detecção de frentes emergentes de pesquisa a partir da análise de redes de colaboração e citação construídas a partir de dados da literatura científica.

Assim, os objetivos a alcançar a caminho de uma solução para o problema (1) são:

i. Análise dos softwares disponíveis para estudo de redes construídas a partir de dados da literatura científica, em especial, do software Sci^2 Tool:

   a) compilação de um conjunto de registros de produções científicas provenientes de diversas bases de dados, que seja objeto de uma análise bibliométrica por *humano*, de forma que possa ser utilizado para realização de *benchmarks* dos aplicativos analisados.

ii. Implementação de melhorias:

   a) Implementação de extensões, com algoritmos e métodos mais avançados, para os programas existentes, de forma que possam atender às especificações (1.a-g); e/ou

   b) implementação de novas ferramentas, para análise de redes científicas, que atendam às especificações (1.a-g); e

   c) elaboração de ampla documentação dos programas implementados.

iii. Testes e distribuição dos programas implementados em versão estável, funcional e documentada.

Na linha de soluções para o problema (2), temos os seguintes objetivos:

iv. Identificar, através de métodos quantitativos de análise cientomtétrica combinados com avaliação de especialistas, dentre os sistemas de reator e ciclo de combustível selecionados pela lista do GIF como os mais promissores, os que realmente mais se desenvolveram nos últimos 10 anos.

a) Identificar, dentre os sistemas mais evoluídos aqueles que, tendo em vista suas necessidades e a base tecnológica do Brasil, poderiam ser considerados os mais adequados para nossos esforços de pesquisa e desenvolvimento.

b) Elaboração de amplo registro da metodologia desenvolvida.

A FIG.1 apresenta um fluxograma que proporciona uma visão mais panorâmica de todas as etapas dos caminhos apontados para a solução dos problemas propostos.

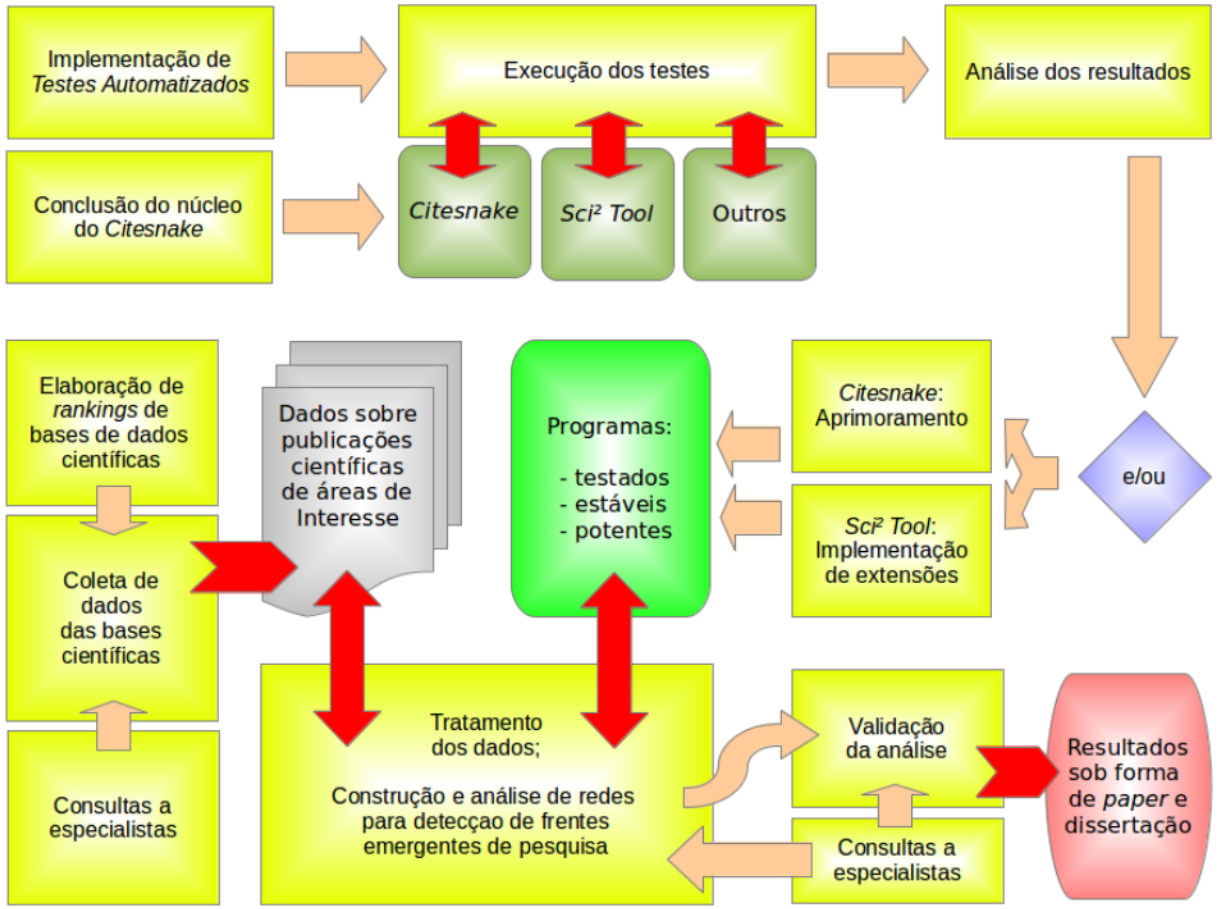

FIGURA 1 – Fluxograma do trabalho realizado para a solução dos problemas levantados

3 SOFTWARES

3.1 Fundamentos teóricos para o *Citesnake*

O desenrolar de projetos na linha de *Métodos para Detecção de Frentes Emergentes de Investigação Científica* conduzidos nos anos de 2011 e de 2012 pelo nosso grupo de pesquisa fez notar a ausência de softwares que realizassem um correto tratamento de dados provenientes de bases de literatura científica, ausência esta que se tornou um obstáculo considerável para o empreendimento de estudos de detecção de frentes emergentes de pesquisa. O aplicativo até então mais adequado para a condução de tais estudos era o *Citespace* [47] que, entretanto, apresentava alguns problemas que dificultavam substancialmente as pesquisas desejadas:

- falhas de segmentação quando do procesamento de alguns arquivos de entrada que, no entanto, atendiam aos padrões do software;
- a compilação de um banco de dados *MySQL* em tempo de execução quando processava os arquivos de entrada, uma operação lenta e que tornava as consultas aos registros processados bastante demoradas;
- a utilização do software por um usuário padrão era bastante custosa, dada a interface não intuitiva do programa;
- impossibilidade de *cruzamento de dados* de bases científicas diferentes: qualquer tentativa neste sentido resultava numa lastimável *confusão de entidades*, ou seja, confusão entre nomes de autores, nomes de instituições, duplicação de nós numa mesma rede de colaboração, entre outras falhas, que impossibilitavam por definitivo o uso simultâneo dos registros de dois ou mais bancos científicos.

Um software ideal para os propósitos dos nossos projetos deveria, portanto, (i) tratar as entidades (objetos e suas classes) adequadamente; (ii) ser eficiente do ponto de vista da complexidade de tempo de execução e espaço de memória utilizada; (iii) ser transparente nos requisitos para processar arquivos e comunicar os erros de execução adequadamente; e (iv) ser simples e intuitivo, de forma que pudesse ser usado por um “usuário médio” dedicado à área.

Assim, o primeiro questionamento a ser feito era: é possivel, do ponto de vista teórico, um software capaz de realizar o *cruzamento* de dados de bases diferentes, ou seja, executar a leitura dos dados, desambiguar informações diferentes e sintetizar informações de mesmo valor, de forma que não haja confusão de objetos e classes? Ora, um tal aplicativo seria um *tradutor*, que realizaria a tradução dos diversos “idiomas” dos bancos de dados científicos para um idioma final, através do qual a informação pudesse ser tratada sem ambiguidades ou multiplicidade de significantes. Do ponto de vista computacional, um tradutor é um autômato que processa palavras pertencentes a uma determinada liguagem e, através das mudanças de estado, as traduz em seus correspondentes pertencentes a uma outra linguagem. A existência de tal autômato define se as linguagens em questão são *regulares*; assim, no sentido inverso, é a regularidade de tais linguagens que determinam a possibilidade de existência de um autômato tradutor [48].

A possibilidade do software desejado estava baseada, então, na possibilidade de demonstrar-se que as linguagens de exportação dos diversos bancos de dados científicos eram, de fato, linguagens regulares. E esssa demonstração não é difícil, apesar de um tanto longa: vamos demonstrar que, para uma linguagem de exportação de dados de uma base científica usual, existe um *autômato não determinístico* que a reconhece.

Uma base de dados científica usual como Web of Science [17], Scopus [18], entre outras, exporta registros em formato *csv* (*comma separated values*, valores separados por vírgulas); nesse formato, existe um *caractere separador*, geralmente a vírgula, podendo ser outro. Sem perda de generalidade, vamos supor que este arquivo possua apenas duas colunas, “A” e “B”, contendo cada uma dada informação de interesse sobre os registros de publicações exportados pelo banco de dados, aos quais corresponderiam as linhas do arquivo exportado. Se existir um autômato que reconheça as palavras contidas na coluna “A” e um autômato que reconheça as palavras contidas na coluna “B”, então o autômato que reconhece cada linha do arquivo é a concatenação destes dois autômatos, sendo que se passa de um estado final do primeiro autômato para o estado inicial do segundo autômato através do caractere separador do formato *csv*. Assim, o problema se reduz a demonstrar que as *strings* contidas numa dada coluna do arquivo exportado são reconhecidas por um autômato não determinístico associado àquela coluna.

De fato, numa célula de uma dada coluna, estão nomes (palavras de um idioma), abreviações ou algarismos decimais, combinados e separados por caracteres específicos. Assim, como num idioma humano, a quantidade de palavras formais é grande porém finita, a linguagem dos nomes e abreviações é regular. Do ponto de vista dos algarismos decimais, se consideramos qualquer um deles como pertencente a $[0-9]^*$, então também pertencem a linguagem regular. Assim, unindo os autômatos dessas linguagens pelos caracteres separadores específicos, temos um autômato capaz de reconhecer as *strings* da coluna. A questão da ordem da combinação destes autômatos e das palavras com erros de digitação é facilmente resolvida pelo seu carácter *não determinístico*. Fica, então, demonstrado que as *strings* contidas numa dada coluna do arquivo exportado constituem uma linguagem regular, e portanto, que a linguagem de exportação de dados de uma base cietífica usual é regular.

Dessa forma, do ponto de vista teórico, o software que nós desejávamos podia ser um *tradutor* para os "idiomas" dos diversos bancos de dados científicos. Do ponto de vista de implementação, optou-se por empregar as *regexes* [49], expressões capazes de representar linguagens pertencentes a um conjunto que contém o conjunto das linguagens regulares. A linguagem de programação escolhida para a implementação das *regexes* foi a linguagem *Python*, na qual uma série de scripts em uso nos projetos do nosso grupo de pesquisa já haviam sido desenvolvidos.

Foram então, inicialmente, implementados alguns scripts em linguagem *Python* para tratar os registros a serem processados a partir do ponto de vista das *regexes*. Os resultados imediatos foram bastante impressionantes:

- o emprego de *expressões regulares* adequadas permitia uma limpeza eficiente dos registros, ou seja, deixou-se de guardar *tudo* em uma base de dados monstruosa, e passou-se a armazenar, em estruturas de dados otimizadas, *apenas* o que era *necessário* aos estudos de detecção de frentes emergentes de pesquisa;

- o uso de estruturas de dados adequadas (como *tabelas de hashing*) para esse armazenamento possibilitava guardar as informações úteis dos registros processados em memória física, de modo que não era necessário o uso de memória de disco, algo que proporcionava a

eficiência desejada às consultas dos registros processados;

- a capacidade de seleção das expressões regulares utilizadas evidenciou o potencial dos scripts para realizar o *cruzamento* de dados proveninetes de diferentes bancos de dados, algo que até então os aplicativos disponíveis, mesmo que se propusessem essa tarefa, não conseguiam realizar.

Desses scripts nasceu um programa que foi batizado de *Citesnake* [53], cujo núcleo passou então a ser aprimorado e cujo desempenho deveria ser confrontado com a performance de outros softwares disponíveis para o tratamento de dados provenientes de bases de literatura científica, como por exemplo, o programa *Sci² Tool*, da Universidade de Indiana que, até então, era apontado pelos especialistas da área como a ferramenta mais promissora em estágio mais avançado de desenvolvimento (vale observar que Citespace e Sci² Tool foram desenvolvidos pelo mesmo grupo de pesquisa [26]). O programa Sci² Tool foi implementado em Java™, e faz uso de uma série de classes da plataforma *CIShell – Cyberinfrastructure Shell* [55] – da Universidade de Indiana. Trata-se de uma plataforma para integração e utilização de algorítmos, bancos de dados, ferramentas e recursos computacionais. Esta plataforma é a base do bem-conhecido software para análise de redes sociais *Network Workbench* [56].

3.2 Exploração dos softwares existentes e opção pelo Citesnake

Não encontramos softwares, além do Sci² Tool, que se propusessem, como o Citesnake, a cruzar e analisar dados provenientes de diferentes bases científicas. Logo, nossas análises se concentraram nesse aplicativo, que seria então o único "concorrente direto" do Citesnake. Este cruzamento seria, de fato, a tarefa mais difícil para um software que não apresentasse uma grande quantidade de falhas de segmentação, como o Citespace, ou seja: já não nos importaríamos com softwares que não conseguissem sequer processar dados sem erros (ou que não comunicassem adequadamente o motivo dos erros), mas somente com aqueles que, possuindo um correto tratamento de erros, se propusessem uma análise eficiente dos dados processados. Os únicos softwares conhecidos nessas condições eram, então, o Citesnake e o Sci² Tool.

Entramos em contato com desenvolvedores da Universidade de Indiana, pedindo esclarecimentos sobre como o software poderia ser usado para cruzar registros provenientes de bases científicas diversas. Para o teste, a intenção era cruzar os dados sobre redes de colaboração entre autores a partir registros em arquivos provenientes apenas das bases de dados Web of Science [17] e Scopus [18].

Em resposta, foi explicado que, uma vez construídas as redes de colaboração a partir dos arquivos de cada base de dados, essas redes deveriam ser intercaladas e, dado que os formatos dos nomes dos autores em cada base são armazenados de forma diferente, deveria ser utilizado o algorítmo *DetectDuplicateNodes* para colapsar nós que representassem um mesmo autor.

Seguimos as intruções e imediatamente ficaram demonstradas as deficiências do algorítmo *DetectDuplicateNodes*:

- o algorítmo colapsa nós cujas etiquetas são semelhantes; essa semelhança é definida por um cálculo simples de semelhança entre *strings*;
- este cálculo necessita que o usuário forneça um *treshold* para o programa, de forma a serem colapsados apenas nós cujas etiquetas apresentem semelhança maior que este *treshold*;
- quando se trabalha com centenas ou milhares de registros, é praticamente impossível acertar um *treshold* tal que não haja falsos positivos ou falsos negativos; assim, o algorítmo *DetectDuplicateNodes* se torna *probabilístico* e, portanto, apresenta uma *taxa* de acerto, enquanto o algorítmo do Citesnake procura ser *determinístico*: o Citesnake faz uso de autômatos *tradutores determinísticos* que procuram traduzir para uma linguagem comum as linguagens das diferentes bases de dados. Ou seja: a menos de casos patológicos previamente conhecidos (que procuramos minimizar) ou de informação errada armazenada, o Citesnake *sempre* vai acertar quais nós deverão ser colapsados.

Uma vez constatada a superioridade do Citesnake em cruzar os dados de diferentes bases, dispensou-se grande quantidade de esforços para consolidar este programa

como uma plataforma bem estabelecida para a condução de estudos de frentes emergentes de pesquisa e buscou-se aprimorá-lo cada vez mais. Tendo como objetivo a excelência do código resultante, sua implementação procurou seguir os seguintes eixos de confecção de software:

1. Orientação a objetos: otimiza o código, deixando-o mais claro e deixa o programa naturalmente preparado para a *persistência de objetos*, ou seja, para a construção facilitada de um banco de dados a partir dos objetos presentes na memória física, caso esta ameace se esgotar. Além disso, a orientação por objetos é decisiva para a possibilidade de implementação de uma interface gráfica de usuário e para assegurar a *modularidade* do Citesnake.

2. Constante aprimoramento das expressões regulares tradutoras: o principal diferencial do Citesnake é traduzir para uma "linguagem comum" as diversas linguagens utilizadas pelas bases de dados científicas para o registro dos dados de um artigo científico; um grande avanço trazido pelo Citesnake foi o uso de expressões regulares construídas a partir da representação hexadecimal de caracteres, o que possibilitou ao programa reconhecer um alfabeto mutíssimo mais amplo: caracteres matemáticos complexos e nomes de autores escandinavos, por exemplo, já não produzem erros; de fato, após a última atualização de expressões regulares, não conhecemos nenhum idioma ou linguagem ocidental que o Citesnake não possa tratar corretamente.

3. Construção de módulos: cada novo módulo é uma nova "linguagem" que o Citesnake consegue traduzir. Essa *modularidade* confere ao nosso programa uma agilidade impressionante: se uma base de dados modificar os padrões com que armazena seus registros, basta modificar apenas um módulo do Citesnake, sem a necessidade de manutenção do núcleo ou dos outros módulos do programa.

3.3 Conclusão do Citesnake beta 4

Em julho de 2014, concluiu-se a versão *beta 4* do Citesnake, disponível para sistemas Microsoft Windows e Unix. Essa versão possui módulos que permitem tratar dados das bases de artigos científicos mais importantes no mundo:

- Web of Science [17];
- Scopus [18];
- PubMed [19];
- Google Scholar [21] (através do uso do programa *Publish or Perish* [50], de A. W. Harzing).

O Citesnake beta 4 foi disponibilizado para *download* em uma página eletrônica de endereço <www.citesnake.org>, no formato de pacote do tipo *zip*; este pacote contém os arquivos:

- **citesnake-base.py**: arquivo de base do Citesnake, no qual está escrito o núcleo de seu código;
- **csnake_wos_mod.py**: arquivo que contém o código que permite ao Citesnake tratar os arquivos de registros de publicações exportados pela base Web of Science;
- **csnake_scp_mod.py**: arquivo que contém o código que permite ao Citesnake tratar os arquivos de registros de publicações exportados pela base Scopus;
- **csnake_gsc_mod.py**: arquivo que contém o código que permite ao Citesnake tratar os arquivos de registros de publicações exportado pelo *site* Google Scholar;
- **csnake_pmd_mod.py**: arquivo que contém o código que permite ao Citesnake tratar os arquivos de registros de publicações exportado pela base PubMed;
- **LEIAME.txt**: arquivo que contém instruções em Português para utilização do Citesnake beta 4;

- **README.md**: arquivo que contém instruções em Inglês para utilização do Citesnake beta 4;

Além destes arquivos, integram também o pacote de distribuição do Citesnake quatro arquivos de exemplos de registros de publicações exportados pelas bases de dados com as quais o Citesnake pode trabalhar:

- **colombeau-savedrecs.csv**, exportado da Web of Science, localizado no diretório **data_from_wosc**;
- **colombeau-scopus.csv**, exportado da Scopus, localizado no diretório **data_from_scop**;
- **colombeau-PoPCites.csv**, exportado do Google Scholar, localizado no diretório **data_from_gsch**;
- **colombeau-pubmed.csv**, exportado da PubMed, localizado no diretório **data_from_pmed**;

Para a confecção destes arquivos de exemplo, exportou-se uns poucos registros de publicações sobre a *Teoria das Funções Generalizadas de Colombeau*, um tema da área de Matemática.

O objetivo da inclusão destes arquivos de exemplo no pacote de distribuição do Citesnake beta 4 é proporcionar ao usuário a possibilidade de comparar os arquivos de registros de seu interesse com arquivos já experimentados no Citesnake se, por ventura, o Citesnake retornar algum erro de leitura de arquivos de entrada. De fato, como as bases de dados científicos frequentemente alteram pequenas características do formato de exportação de seus registros, é conveniente que o usuário possa comparar arquivos que o Citesnake pode ler contra arquivos que o programa já não consegue ler, de forma que, através de alguma pequena edição do próprio usuário no novo arquivo de registros exportado, o Citesnake consiga processá-lo, sem a necessidade de permanecer aguardando a atualização de algum módulo.

Como a implementação do Citesnake constituiu o núcleo substancial deste trabalho, todo o seu código - da base e dos módulos - é apresentado nos apêndices desta dissertação.

Após processar os registros proveninentes das bases de dados citadas, o Citesnake beta 4 exporta uma *meta-rede* contendo cinco redes de dados sobre publicações científicas:

- rede Artigo x Autor;
- rede Artigo x Palavra-Chave;
- rede Artigo x Referência;
- rede Artigo x Ano de publicação;
- rede Autor x Instituição.

Essas cinco redes, após análise em software adequado, fornecem informações sobre redes de colaboração científica por tema, por instituição, por localidade e por ano, além de diversos outros dados de interesse do usuário. A título de exemplo, através de aritmética matricial simples, podemos compor rapidamente redes *monomodais*, cuja natureza da relação dos nós seja de interesse como, por exemplo, citação das mesmas referências, trabalhos envolvendo os mesmos temas-chaves, co-autoria em publicações, filiação a um mesma instituição, entre outras; essas redes monomodais podem ser clusterizadas com o emprego de diversos algorítmos apropriados permitindo ao analista identificar agrupamentos úteis ao trabalho de detecção de tendências emergentes dentro de um determinado campo do conhecimento. Além disso, a opção por tratar o *tempo* como *objeto* (entidade “Ano de publicação”), e não como *atributo*, por simples que pareça, proporciona grande agilidade ao analista para investigar as características das relações no tempo, sem a necessidade do emprego de algorítmos mais específicos e complexos.

A meta-rede é exportada no formato padrão do ORA [51], o principal software para análise de redes sociais até então em uso no nosso grupo de pesquisa. Adicionalmente, é possível armazená-la em memória como um conjunto de cinco grafos tratáveis atraves do pacote *NetworkX* [52], desenvolvido para o estudo de redes complexas. Por fim, praticamente não existem limites para a quantidade de registros que o Citesnake pode processar: as limitações ficam por conta do hardware utilizado.

3.4 O Citesnake e os scripts de Leydesdorff

Após a conclusão do Citesnake beta 4, tomamos conhecimento da existência de alguns scripts disponibilizados por Loet Leydesdorff em [62] que tornam possível a conversão de arquivos exportados por Scopus para o formato em que são exportados os dados da Web of Science; também existem nesta página eletrônica scripts para tratar dados exportados do Google Scholar. Lenvantou-se, então, um questionamento natural: teria sido o Citesnake já superado por um conjunto de scripts que cruza dados de diversas bases? A resposta para essa pergunta é *não*. Testamos os scripts de Leydesdorff com os próprios arquivos de exemplo disponibilizados no pacote de distribuição do Citesnake, e constatamos que eles realizam bem o que se propõem: colocam os dados exportados por diferentes bases em um formato comum (Microsoft Access ou dBase). Todavia, estes scripts não cruzam os dados provenientes de bases diferentes de forma automática, como faz o Citesnake: o trabalho de detectar duplicidades de registros é deixado para o usuário realizar manualmente, o que significa, na prática, o retorno aos “malabarismos” com grandes quantidades de dados. O Citesnake, portanto, continua a trazer uma novidade sem precedentes: a automatização do cruzamento de dados de diversas bases. Além disso, não existe na página de Leydesdorff qualquer script que se aplique a tratar dados exportados da PubMed; já o Citesnake possui um módulo que lhe permite trabalhar com eficácia arquivos de registros desta base de dados.

Parte II

MÉTODOS PARA DETECÇÃO DE FRENTES EMERGENTES DE PESQUISA

4 MÉTODOS

Nesta segunda parte do nosso projeto, procurou-se identificar, através do software desenvolvido e apresentado na Parte I, dentre aqueles sistemas selecionados como os mais promissores na lista do GIF, os que realmente mais se desenvolveram nos últimos 10 anos: pretendeu-se, dessa forma, contribuir para a consolidação de um método eficaz de condução de estudos de detecção de frentes emergentes de pesquisa a partir da análise de redes de colaboração e citação, redes estas construídas a partir de dados da literatura científica. Procedeu-se, então, à aplicação do Citesnake no desenvolvimento de uma metodologia para a condução de estudos sobre tendências e frentes emergentes de pesquisa. Tal metodologia foi aplicada à área de reatores nucleares e ciclos de combustível avançados. Por primeiro, procuramos nos ater à lista consolidada pelo GIF, constituida por seis conceitos de reatores inovadores:

- *Very High Temperature Reactor* (VHTR), ;
- *Sodium-cooled Fast Reactor* (SFR);
- *Supercritical Water Reactor* (SCWR);
- *Gas-cooled Fast Reactor* (GFR);
- *Lead-cooled Fast Reactor* (LFR);
- *Molten Salt Reactor* (MSR).

A partir desta lista, procuramos delinear o estágio de desenvolvimento de cada um destes seis tipos de reatores, bem como aqueles que com mais veemência têm emergido como opções consolidadas.

Através de consulta a especialistas, foram determinadas algumas palavras-chave que desempenhariam o papel de *marcos técnico-científicos* da evolução dos conceitos. Para um momento inicial de avaliação do método, foram selecionados apenas dez marcos: termos simples que deveriam denotar etapas bem definidas no progresso de cada tipo de reator, desde os esforços de pesquisa e desenvolvimento até o comissionamento. Os marcos selecionados foram:

- “*Conceptual design*”;
- “*Operating experience*”;
- “*Technology projects*”;
- “*Test facility*”;
- “*First of a kind engineering* (FOAKE)”;
- “*Fuel performance assessment*”;
- “*Proof of concept demonstration*”;
- “*Phenomena identification and ranking tables* (PIRT)”;
- “*Code scaling, uncertainty, and analysis* (CSAU)”;
- “*High burn-up demonstration*”.

Foram realizadas buscas em bancos de dados científicos por registros que tratassem de cada um dos seis tipos de reatores inovadores em questão e que apresentassem ou não os marcos selecionados. Dentre as bases de dados cujo “idioma” o Citesnake é capaz de tratar, escolhemos utilizar Web of Science, Scopus e Google Scholar; PubMed foi descartada porque considerou-se que um banco de dados dedicado à área médica e afins não deveria conter uma quantidade substancial de artigos relacionados a reatores nucleares avançados.

4.1 Análise quantitativa das publicações

Para um reconhecimento preliminar da área de pesquisa científica a ser investigada, foram compostos gráficos das quantidades de publicações envolvendo cada tipo de reator por ano, desde 2000 - quando a iniciativa do GIF foi consolidada - até o final de 2012. A produção científica de 2013 não foi incluída nesta análise pois, até o momento da coleta de dados para este trabalho, realizada entre novembro de 2013 e fevereiro de 2014, era grande a chance de nem todas as publicações terem sido indexadas; desse modo, a adição dos dados da produção científica de 2013 ao conjunto de dados analisados poderia provocar uma falsa idéia de declínio do número de trabalhos envolvendo algum tipo de reator avançado.

A FIG.2 mostra o número de publicações, por ano e por tipo de reator, indexadas por Web of Science e Scopus. Este gráfico traz o número de artigos que fazem referência no título, no resumo ou nas palavras-chave, a cada um dos seis tipos de reatores estudados; os artigos contados neste gráfico não contêm necessariamente algum dos marcos técnico-científicos apresentados anteriormente: dessa forma, pode-se detectar produção científica concernente a estágios bastante preliminares ou mesmo "pioneiros" do desenvolvimento de cada conceito de reator inovador.

A FIG.3 apresenta também o número de publicações, por ano e por tipo de reator, indexadas por Web of Science e Scopus mas, agora, apenas aquelas que mencionam pelo menos um dos marcos técnico-científicos selecionados.

Por fim, a FIG.4 mostra o número de publicações, por ano e por tipo de reator, indexadas por Web of Science, Scopus e Google Scholar, que podem conter ou não conter referência a algum dos dez marcos técnico-científicos da evolução dos conceitos. É oportuno observar que este gráfico nos proporciona uma idéia do estágio de desenvolvimento de cada conceito de reator em um "*mundo mais informal*": o Google Scholar, através dos seus algorítmos avançados de indexação da web, armazena uma série de artigos e documentos que ainda não foram indexados pelas grandes bases científicas, ou mesmo que sequer foram publicados - de fato, alguns pesquisadores disponibilizam em seus *sites* pessoais vários *preprints* de trabalhos submetidos a periódicos científicos; dessa forma, ferramentas como o Google Scholar nos permitem ter contato com o atual estágio das "fronteiras mais quentes" de uma determinada área de pesquisa.

4.2 Análise das redes de publicações

Vamos apresentar uma análise, do ponto de vista de redes complexas, restrita aos registros indexados por Web of science e Scopus que referenciassem pelo menos um dos seis tipos de reatores ou no título, ou no resumo ou entre as palavras-chave: esta análise já proporciona uma apresentação efetiva da metodologia desenvolvida. Pretendemos estudar conjuntos maiores de publicações em trabalhos futuros.

Utilizando o Citesnake, a partir do conjunto de registros descrito acima,

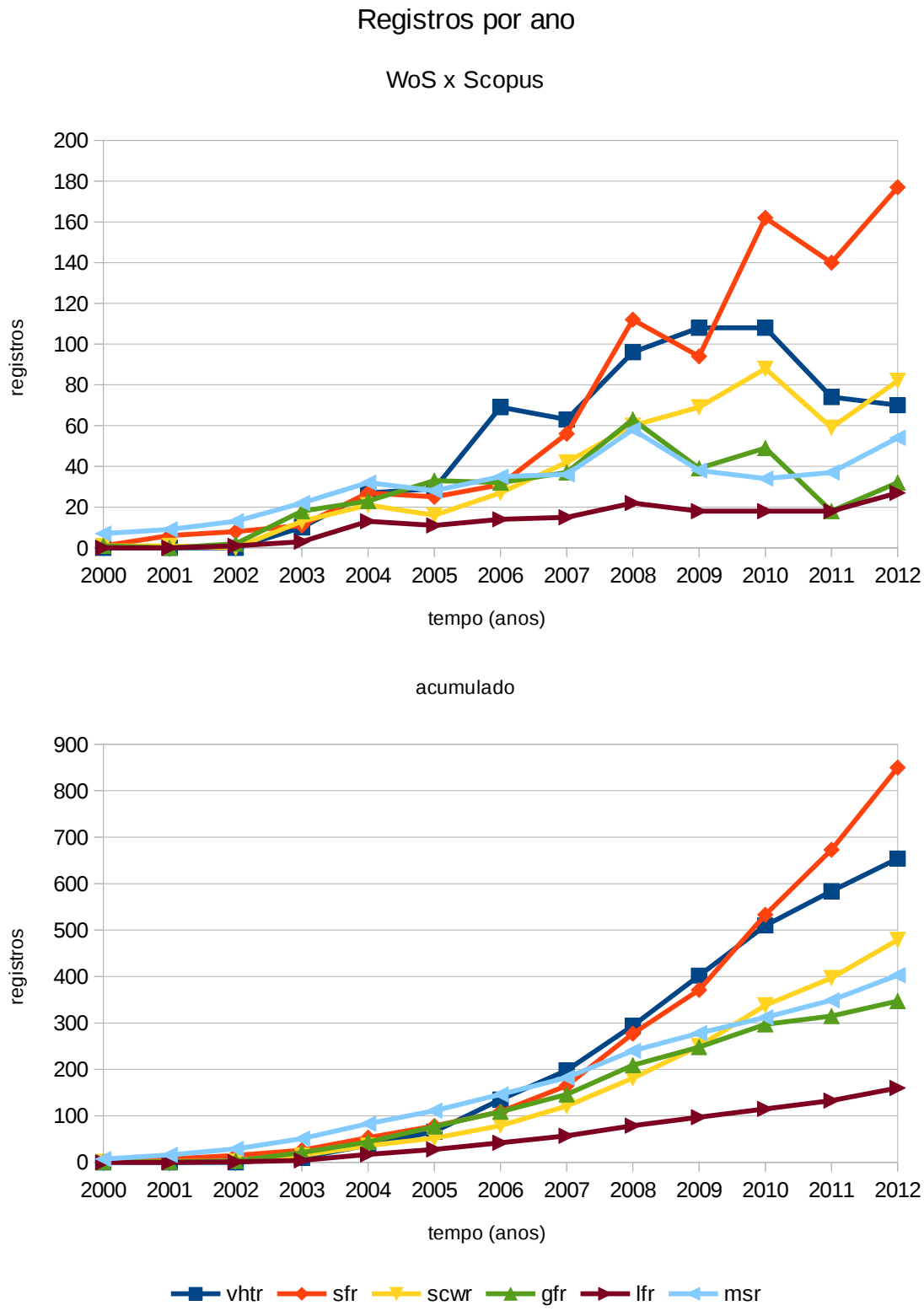

FIGURA 2 – Número de registros por ano, segundo Web of Science e Scopus, referenciando no título, no resumo ou nas palavras-chave, cada um dos seis tipos de reatores estudados, e não necessariamente *mencionando algum dos dez marcos técnico-científicos da evolução dos conceitos de reatores inovadores*

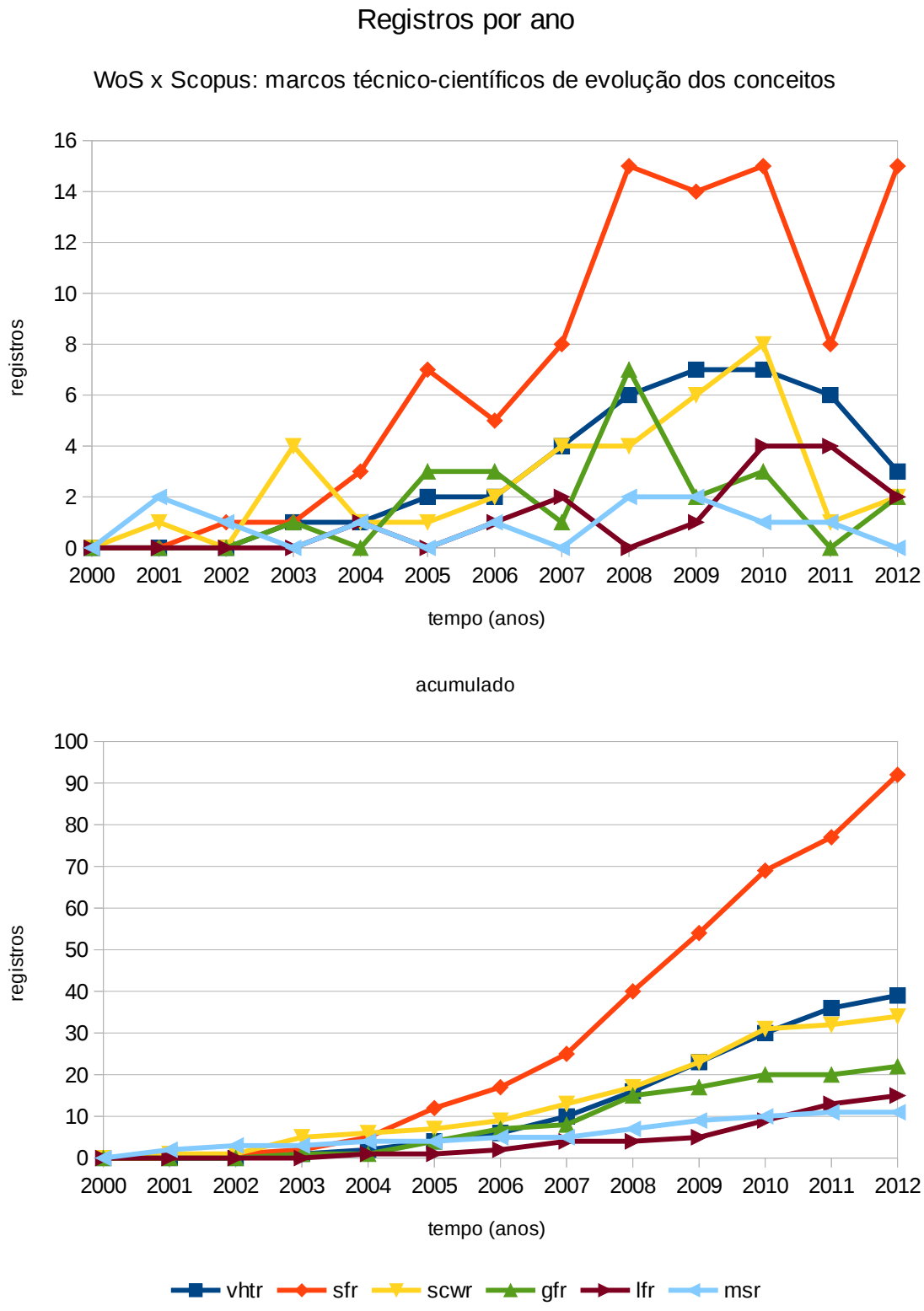

FIGURA 3 – Número de registros por ano, segundo Web of Science e Scopus, referenciando no título, no resumo ou nas palavras-chave, cada um dos seis tipos de reatores estudados, e necessariamente *mencionando algum dos dez marcos técnico-científicos da evolução dos conceitos de reatores inovadores*

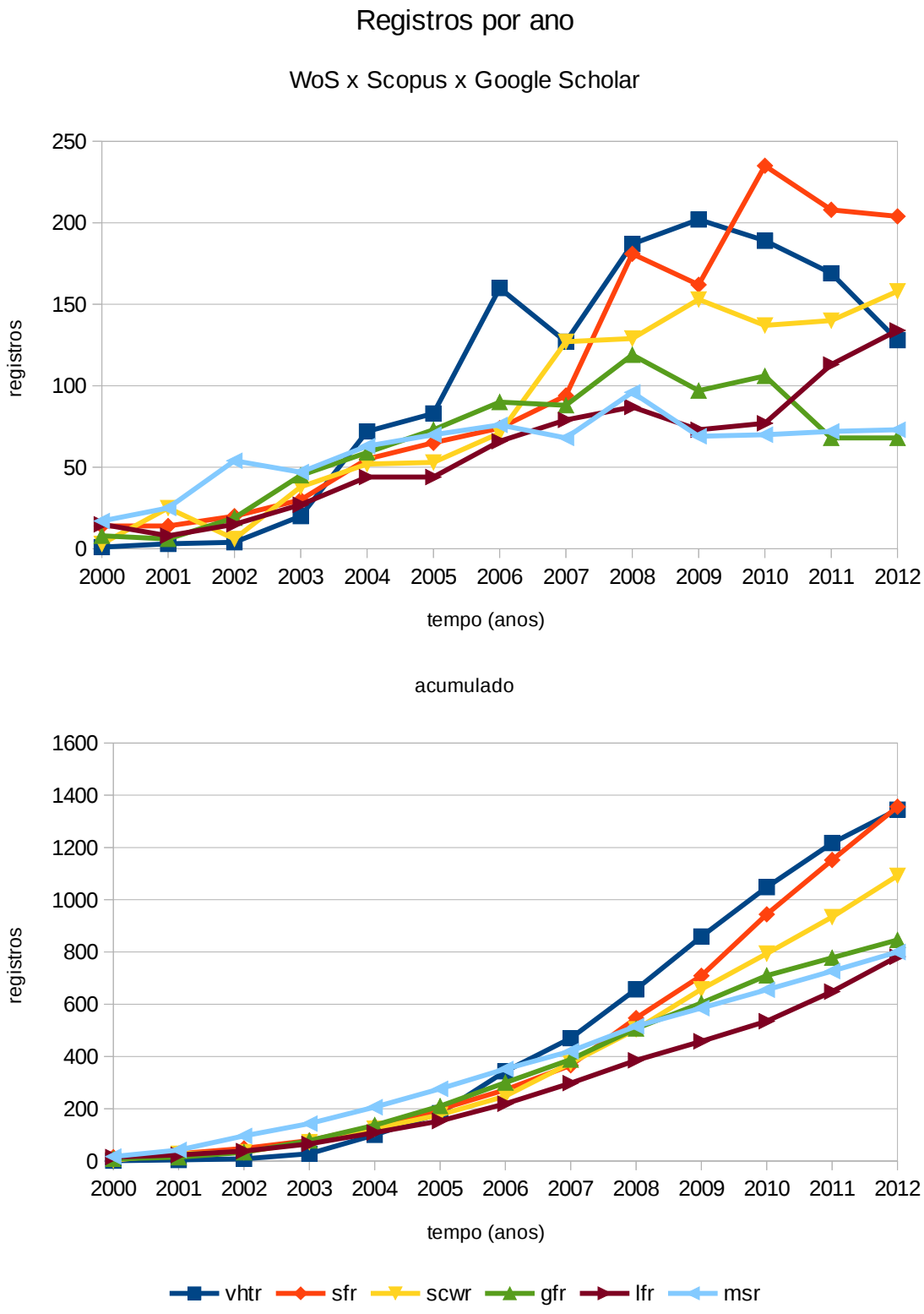

FIGURA 4 – Número de registros por ano, segundo Web of Science, Scopus e Google Scholar, referenciando no título, no resumo ou nas palavras-chave, cada um dos seis tipos de reatores estudados, e não necessariamente *mencionando algum dos dez marcos técnico-científicos da evolução dos conceitos de reatores inovadores*

produziu-se uma meta-rede contendo *nove redes*: as cinco redes naturalmente geradas pelo Citesnake; uma rede especial, denominada “Artigo x Conceito de reator”, gerada através de um módulo para o Citesnake implementado especialmente para este estudo; e três redes obtidas através da operação de multiplicação de uma matriz por sua transposta (*folding*):

- rede Artigo x Artigo;
- rede Autor x Autor;
- rede Palavra-chave x Palavra-chave;

Uma vez geradas as redes, nós calculamos as principais medidas do ponto de vista de redes, com o objetivo de detectar as entidades chave de cada rede e as frentes de pesquisa a elas relacionadas.

5 RESULTADOS

5.1 Instituição x autor

Através do cálculo da *centralidade de grau* de cada instituição, foram selecionadas as mais ativas, ou seja, aquelas que agrupam mais autores que publicaram artigos na área de reatores avançados. O resultado é apresentado na TAB.1.

É interessante apontar que todas estas instituições têm trabalhos publicados envolvendo todos os seis conceitos de reatores inovadores de interesse, com execeção de “INDIRA GANDHI CTR. ATOM. RES.”, que não apresenta publicação acerca dos conceitos VHTR e LFR.

5.2 Autor x artigo

Com base na centralidade de grau de cada autor, nós detectamos os mais ativos, isto é, aqueles que apresentaram mais publicações no campo de pesquisa analisado, e os conceitos de reatores nos quais têm trabalhado. A TAB.2 apresenta

TABELA 1 – As dez instituições com maior número de autores ativos na área analisada

Instituições	**Grau**
KOREA ATOM ENERGY RES INST	461
JAPAN ATOM ENERGY AGCY	364
CEA	229
INDIRA GANDHI CTR ATOM RES	177
IDAHO NATL LAB	146
XI AN JIAO TONG UNIV	137
MIT	113
ARGONNE NATL LAB	108
PAUL SCHERRER INST	101
UNIV WISCONSIN	97

TABELA 2 – Os dez autores com maior número de publicações no campo analisado

Autor	**Grau**	**Conceito(s)**
Y. KIM	56	VHTR, SFR, LFR
K. MIKITYUK	46	SFR, GFR, LFR, MSR
H. OHSHIMA	43	SFR, GFR
S. KIM	42	VHTR, SFR, SCWR, LFR
I. PIORO	40	SFR, SCWR, LFR, MSR
X. CHENG	38	SFR, SCWR
R. CHAWLA	37	SFR, SCWR, GFR
P. HEJZLAR	33	SFR, GFR, LFR
H. YAMANO	30	SFR, LFR
P. V. TSVETKOV	29	VHTR

o resultado desta análise.

Podemos, então, ordenar os tipos de reatores por frequência nos trabalhos destes dez autores, do mais frequente para o menos frequente: SFR, LFR, SCWR, GFR, VHTR e MSR.

5.3 Autor x autor

Na rede *autor x autor*, dois autores estão conectados se foram co-autores de uma mesma publicação. A FIG.5 mostra os autores que repetidamente aparecem entre os mais bem colocados após ordenação pelas medidas de análise de redes sociais disponíveis no aplicativo ORA, utilizado para realizar as análises aqui apresentadas. A relação e descrição dessas medidas aplicáveis a redes sociais pode ser encontrada em [57]. O Valor apresentado na FIG.5 é a porcentagem de medidas que classificam o autor entre os três primeiros. Em palavras simples, esses autores podem ser vistos como os mais colaborativos.

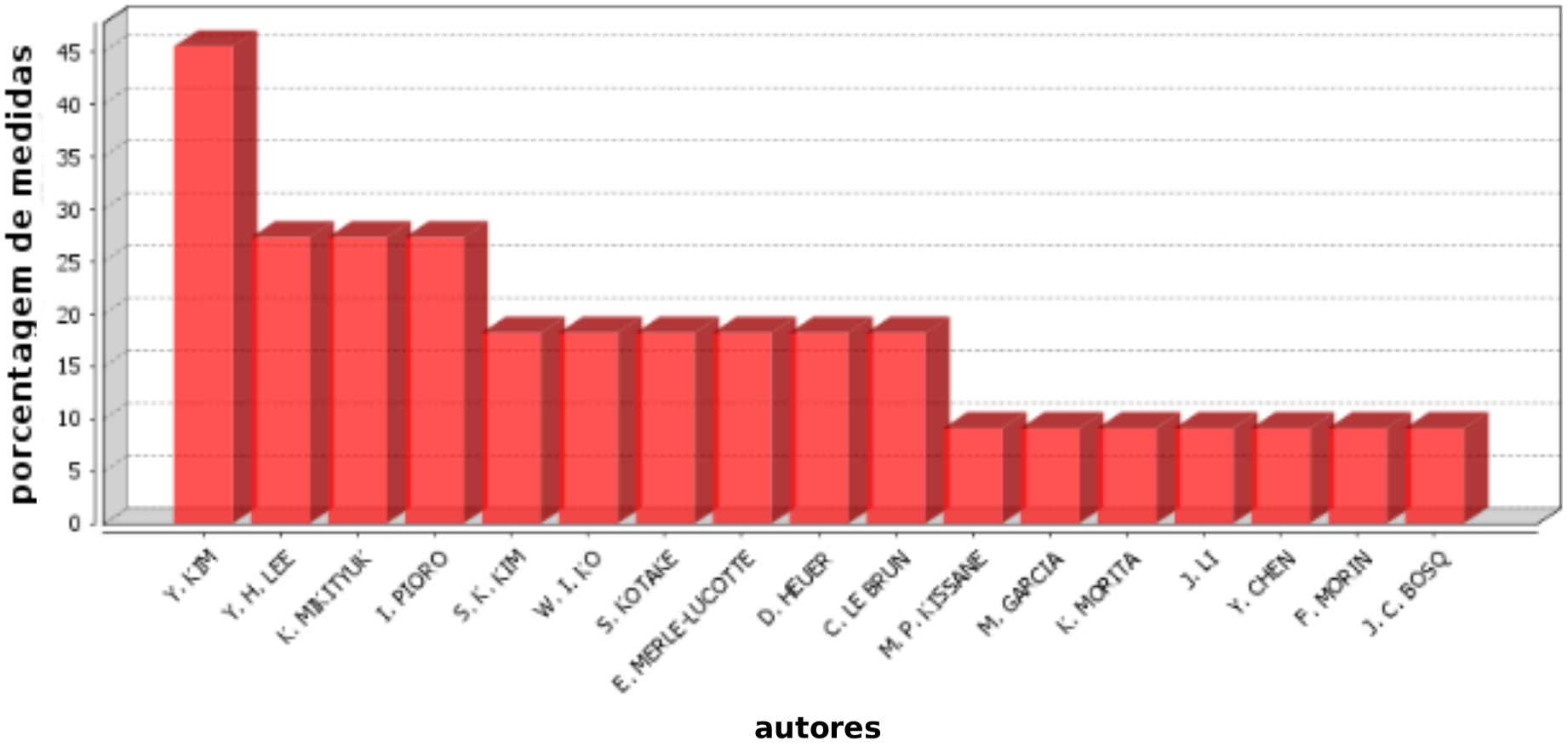

FIGURA 5 – Autores recorrentemente bem classificados quanto às medidas usuais para redes de colaboração

TABELA 3 – As dez palavras-chave mais frequentes nas publicações analisadas

Palavra-chave	**Grau**
vhtr	106
sodium-cooled fast reactor	68
molten salt reactor	60
fast reactor	54
scwr	52
hydrogen production	46
corrosion	44
sodium	34
supercritical water	34
heat transfer	33

5.4 Artigo x palavra-chave

O cálculo do grau de cada palavra-chave nos permite detectar as mais populares; as dez mais bem colocadas são apresentadas na TAB.3.

5.5 Artigos x referências

O cálculo do grau de cada referência nos proporciona um resultado bastante insteressante: as referências mais citadas, bem como os conceitos de retores inovadores que cada uma aborda e os conceitos abordados pelas publicações que citam essas referências. Os resultados são apresentados na TAB.4. Note-se que as referências são apresentadas no formato padrão da Web of Science; por estranho que seja, este formato omite o título das publicações.

5.6 Fragmentação das redes monomodais

Na rede *artigo x artigo*, dois artigos estão conectados se abordam pelo menos um tipo de reator em comum. Na rede *palavra-chave x palavra-chave*, duas palavras chave estão conectadas se aparecem juntas em algum artigo. E, conforme

TABELA 4 – As dez referências mais citadas pelos registros analisados

Referência	**Grau**	**Conceito(s)**
YAMAGATA K, 1972, INT J HEAT MASS TRAN, V15, P2575, DOI 10.1016/0017-9310(72)90148-2	35	SCWR
CHANG J, 2007, NUCL ENG TECHNOL, V39, P111	30	VHTR
PIORO IL, 2004, NUCL ENG DES, V230, P69, DOI 10.1016/J.NUCENGDES.2003.10.010	23	SCWR
PIORO I. L., 2007, HEAT TRANSFER HYDRAU	23	SCWR
SQUARER D, 2003, NUCL ENG DES, V221, P167, DOI 10.1016/S0029-5493(02)00331-X	23	SCWR
US DOE NUCLEAR ENERGY RESEARCH ADVISORY COMMITTEE AND THE GENERATION IV INTERNATIONAL FORUM, 2002, TECHN ROADM GEN 4 NU	19	all
SHIMAKAWA Y, 2002, NUCL TECHNOL, V140, P1	18	SFR
MIKITYUK K, 2005, ANN NUCL ENERGY, V32, P1613, DOI 10.1016/J.ANUCENE.2005.06.002	18	GFR
MACDONALD P.E., 2003, INEELEXT0300870	18	VHTR
TORGERSON DF, 2006, NUCL ENG DES, V236, P1565, DOI 10.1016/J.NUCENGDES.2006.04.020	18	SCWR

explicado anteriormente, na rede *autor x autor*, dois autores estão conectados se foram co-autores de alguma publicação.

Para estas três redes monomodais compiladas (artigo x artigo, autor x autor, e palavra-chave x palavra-chave) foi possível calcular seu *índice de fragmentação*: tal índice consiste de um número entre 0 e 1, onde 1 indica uma rede totalmente fragmentada (todos os nós são isolados). A rede mais fragmentada é a rede autor x autor: 62.2% dos nós são desconectados. A rede menos fragmentada é a rede palavra-chave x palavra-chave: 28.1% apenas são desconectados. A rede artigo x artigo apresenta fragmentação de 31.8%.

6 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Observando o que é apresentado na TAB.1, podemos notar que instituições asiáticas parecem liderar o *ranking* do interesse por sistemas nucleares inovadores. Além disso, a FIG.5 permite confirmar que a maioria dos autores recorrentemente mais bem classificados quanto às medidas usuais aplicáveis a redes sociais são os autores ligados a instituições asiáticas: isto pode ser indicador de um especial conservadorismo do setor nuclear no ocidente. De fato, os sistemas do tipo PWR têm sido amplamente explorados ao longo de várias décadas, e tal exploração proporciona confiança a indústrias e governos para investir antes em reatores conservadores que em reatores inovadores. É razoável pensar que as instituiçoes ocidentais estejam se dedicando menos aos novos sistemas nucleares por perceberem uma opção preferencial da indústria por sistemas bem conhecidos e que possam constituir *sistemas modulares* - reatores pequenos de menor porte. Centrais nucleares, com base nesses reatores poderiam ser construídas de forma incremental, reduzindo o risco econômico e as unidades em função de seu tamanho e sistemas passivos de segurança implicariam em áreas de exclusão bem menores, bem como menores consequências no caso de um improvável acidente.

Os resultados das análises apresentadas na seção 5.2 e apontam o SFR como o conceito preferido pelos autores que mais publicam, informação corroborada pelos gráficos apresentados na seção 4.2. Estes mesmos gráficos apontam uma certa “disputa” entre VHTR e SCWR pelo “segundo lugar” quanto à presença em publicações; quantos à preferência dos autores mais prolíferos, o “segundo lugar” parece ser “disputado” pelos conceitos SCWR e LFR. De fato, nos últimos quatro anos do período analisado, observa-se uma tendência de queda do interesse pelo VHTR, tendência esta concomitante com o aumento do interesse pelo LFR: segundo informações bastante recentes publicadas no *site* do GIF [58], é inclusive esperado que a Rússia conclua um protótipo de reator do tipo LFR por volta do ano de 2020. Por outro lado, a palavra-chave mais presente nos artigos publicados na área é “vhtr”, seguida por “sfr”: é possível pensar, diante desses dados, que o VHTR já foi uma opção bastante explorada, mas tende a ser gradativamente abandonada; de acordo com informações do *site* do GIF [58], a China já iniciou a construção de um protótipo

de reator de altas temperaturas chamado *HTR-PM*, o que constituiu um primeiro passo no desenvolvimento de reatores do tipo VHTR.

Quanto às referências mais citadas, em sua grande maioria, tratam do conceito SCWR e foram publicadas na década de 2000, quando a iniciativa do GIF já havia sido implementada. Dessa forma, SFR e SCWR se consolidam como os conceitos de reatores nucleares inovadores que mais vêm atraindo a atenção dos pesquisadores da área na última década. Com efeito, reatores do tipo SFR constituem “uma fonte energética atrativa para nações que desejam fazer o melhor uso de recursos limitados de combustível nuclear e gerenciar rejeitos nucleares através do fechamento do ciclo de combustível” [59]; já a popularidade do conceito SCWR poderia ser creditada à ideia que seu desenvolvimento tecnológico seja algo incremental a partir dos reatores refrigerados a água atuais [60], o que supostamente diminuiria os custos de implementação da nova tecnologia.

Finalmente, a baixa fragmentação da rede de palavras-chave e a alta fragmentação da rede de autores sugerem que os pesquisadores envolvidos na área poderiam trabalhar de forma mais colaborativa com cientistas de outras instituições, de forma que os tópicos estudados, já aparentemente bem consolidados, pudessem se desdobrar em novas tendências de pesquisa. Por outro lado, não é incomum que, ao pesquisar novas tecnologias, as apostas de instituições e até mesmo de países sejam inicialmente bastante segmentadas e mantidas em silos, enquanto patentes são solicitadas e interesses comerciais são lastreados. No caso da indústria nuclear, em que os desenvolvimentos são mais lentos, é de se esperar que este período observado de fragmentação nas redes continue ainda por vários anos.

7 CONCLUSÕES E TRABALHOS FUTUROS

O objetivo principal deste trabalho foi plenamente alcançado: após cuidadosa análise de aplicativos destinados a estudos cientométricos, foi implementado e apresentado o Citesnake, uma ferramenta computacional robusta e confiável para análise de produção científica e detecção de frentes emergentes de pesquisa, pensada para por fim à ausência de softwares, tanto gratuitos como proprietários, que fossem capazes de consolidar, numa mesma rede, dados exportados por bancos científicos diversos, sem confusões ou duplicidade de informações. A robustez do Citesnake ficou suficientemente evidente após o processamento de milhares de regitros exportados por diversas bases de dados em pouco minutos; além disso, a variedade de redes geradas proporcionou material suficiente para um estudo cientométrico profundo da área de Reatores Nucleares Inovadores, de modo que foi necessário reduzir a quantidade de análises a serem conduzidas a partir das redes exportadas, para que o projeto pudesse ser realizado dentro de escopo e prazo de uma dissertação de mestrado. Os recursos do Citesnake o tornam não apenas mais uma alternativa, mas sim uma ferramenta nova e, até onde se sabe, sem precedentes, que se une a tantas outras que pululam as "caixas de ferramentas" dos pesquisadores que se dedicam à "*Ciência da Ciência*", termo usado por alguns para se referir à Cientometria.

Na segunda parte deste trabalho, ao utilizar-se o Citesnake no estudo dos conceitos de reatores inovadores que mais se desenvolveram nos últimos 10 anos, contribuiu-se para consolidação de uma metodologia eficaz para a detecção de tendências emergentes de pesquisa através dos dados de produção científica. As análises realizadas forneceram um panorama da área de Reatores Inovadores compatível com o que foi recentemente publicado pelo GIF na última atualização de seu *Roadmap* Tecnológico [61]. Com efeito, sabemos que nossa contribuição não esgota o assunto: muitas outras redes provenientes dos dados da produção científica podem ser geradas e analisadas, com e sem a ajuda do Citesnake. Fazendo referência à epígrafe desta dissertação, é evidente um grande buraco entre as análises que *podemos* fazer e as análises que já *conseguimos* fazer: a quantidade de informação disponível sobre produção científica, somada à capacidade de processamento dos

computadores atuais, torna possível, como nunca antes, o estudo de como a Ciência se faz (ou como a Ciência *está* se fazendo) em cada campo; entretanto, a nossa capacidade de implementar ferramentas computacionais adequadas, bem como a capacidade de levar a bom termo todas as análises, ainda não acompanham a velocidade com que evoluem computadores e bancos de dados.

Com a intenção de agregar ao Citesnake permanente aprimoramento, já está em trabalho um módulo que permita-o "compreender e traduzir" o "idioma" do *Derwent Innovation Index*, um banco de dados de registro de patentes disponibilzado na web pela Thomson Reuters através da Web of Knowledge [17]. De fato, no início do desenvolvimento do Citesnake já havia interesse em incluir o Derwent entre os bancos de dados cujo dialeto o nosso aplicativo pudesse tratar, mas notava-se uma grande irregularidade na forma como dados eram exportados, ou seja, não se conseguia processar muita coisa sem que uma quantidade considerável de "lixo" permeasse os dados de interesse. Percebe-se hoje que a Thomson Reuters adequou o formato de exportação de registros do Derwent ao padrão utilizado para exportação de dados de outros índices, o que nos proporcionou tornar a desenvolver um módulo para que o Citesnake possa tratar os registros de patentes e cruzá-los com os dados de outras bases.

Há ainda o interesse em incrementar o Citesnake com um recurso que proporcionasse a detecção, nas redes geradas, de *nós pivotais*, ou seja, trabalhos científicos ou autores cujo posicionamento na rede indica que são "caminho" para a emergência de novas tendências ou mesmo para novos paradigmas de pesquisa: a detecção destes nós vem se consolidando como um problema relevante na área de softwares devotados à Cientometria [44]. Pretendemos abordar esse problema através da implementação de um algorítmo para clusterização baseado em lógica fuzzy, que poderia indicar nós com comportamento de pivôs. De fato, não conhecemos, até o momento, qualquer implementação em Python de tal algorítmo, e as bibliotecas em Matlab e C que o implementam possuem graves restrições da quantidade de nós no grafo de entrada.

Por fim, não podemos deixar de abordar a questão natural da implementação de uma interface gráfica de usuário para o Citesnake: nós procuramos conduzir a codificação de tal modo que nosso aplicativo estivesse preparado naturalmente para atuar em conjunto com uma eventual interface gŕafica; em especial, o Cites-

nake é preparado para receber uma interface desenvolvida em *wxPython* [54], uma biblioteca devotada à implementação de interfaces gráficas. Fugindo o desenvolvimento dessa interface ao escopo deste projeto, aguardamos, inclusive com certa inquietação, a oportunidade de tratar esse problema em trabalhos futuros.

Apêndices

APÊNDICE A – CÓDIGO DO CITESNAKE

Código do arquivo **citesnake-base.py**.

```
1  #! /usr/bin/python
2  #
3  # Citesnake - A tool for co-citation and co-authorship
       analysis in python.
4  # Copyright (C) 2011 Jose Sergio Bleckmann Reis Junior
5  #
6  # This work is licensed under the Creative Commons
7  # Attribution-NonCommercial-NoDerivs 3.0 Unported License. To
       view a copy of
8  # this license, visit http://creativecommons.org/licenses/by-
       nc-nd/3.0/ or
9  # send a letter to Creative Commons, 444 Castro Street, Suite
       900,
10 # Mountain View, California, 94041, USA.
11
12
13 import os
14 import re
15 import csv
16 import time
17 import csnake_wos_mod as wos
18 import csnake_scp_mod as scp
19 import csnake_gsc_mod as gsc
20 import csnake_pmd_mod as pmd
21 import networkx as ntx
22 import matplotlib.pyplot as plt
23 import numpy as npy
24
```

```
25
26 fi = 0
27
28 verbose  = True
29
30 treshold = 2
31
32 wos_files_dir = './data_from_wosc/'
33 wos_files_list = os.listdir(wos_files_dir)
34
35 scp_files_dir = './data_from_scop/'
36 scp_files_list = os.listdir(scp_files_dir)
37
38 gsc_files_dir = './data_from_gsch/'
39 gsc_files_list = os.listdir(gsc_files_dir)
40
41 pmd_files_dir = './data_from_pmed/'
42 pmd_files_list = os.listdir(pmd_files_dir)
43
44 papers_array = []
45 authors_array = []
46 key_words_array = []
47 institutions_array = []
48 references_array = []
49 years_array = []
50
51
52 class Paper:
53     node_id = 0
54     title = 'No Title'
55     year = [0, 0]
56     authors_num = 0
57     key_words_num = 0
```

```
58      references_num = 0
59
60      def __init__(self, node_id, title, year_id):
61          self.node_id = node_id
62          self.title = title
63          self.year = [year_id, 1]
64          return
65
66      def set_author(self, nid):
67          if self.authors_num > 0:
68              for i in range(1, self.authors_num + 1):
69                  if self.__dict__['author␣' + str(i)][0] ==
                        nid:
70                      return
71
72          self.authors_num = self.authors_num + 1
73          self.__dict__['author␣' + str(self.authors_num)] = [
                nid, 1]
74          return
75
76      def set_key_word(self, nid):
77          if self.key_words_num > 0:
78              for i in range(1, self.key_words_num + 1):
79                  if self.__dict__['key_word␣' + str(i)][0] ==
                        nid:
80                      return
81
82          self.key_words_num = self.key_words_num + 1
83          self.__dict__['key_word␣' + str(self.key_words_num)]
                = [nid, 1]
84          return
85
86      def set_reference(self, nid):
```

```
87          if self.references_num > 0:
88              for i in range(1, self.references_num + 1):
89                  if self.__dict__['reference ' + str(i)][0] ==
                        nid:
90                      return
91
92          self.references_num = self.references_num + 1
93          self.__dict__['reference ' + str(self.references_num)
               ] = [nid, 1]
94          return
95
96      def set_year(self, nid):
97          if self.year > 0:
98              return
99
100         self.year = [nid, 1]
101         return
102
103
104 class Author:
105     node_id = 0
106     name = 'No Name'
107     institutions_num = 0
108
109     def __init__(self, node_id, name):
110         self.node_id = node_id
111         self.name = name
112         return
113
114     def set_institution(self, nid):
115         if self.institutions_num > 0:
116             for i in range(1, self.institutions_num + 1):
117                 if self.__dict__['institution ' + str(i)][0]
```

```
                    == nid:
118                     self.__dict__['institution ' + str(i)][1]
                            = self.__dict__['institution ' + str(
                        i)][1] + 1
119                     return
120
121         self.institutions_num = self.institutions_num + 1
122         self.__dict__['institution ' + str(self.
              institutions_num)] = [nid, 1]
123         return
124
125
126 class Key_Word:
127     node_id = 0
128     name = 'No name'
129
130     def __init__(self, node_id, name):
131         self.node_id = node_id
132         self.name = name
133         return
134
135
136 class Institution:
137     node_id = 0
138     name = 'No name'
139
140     def __init__(self, node_id, name):
141         self.node_id = node_id
142         self.name = name
143         return
144
145
146 class Reference:
```

```
147     node_id = 0
148     name = 'No name'
149
150     def __init__(self, node_id, name):
151         self.node_id = node_id
152         self.name = name
153         return
154
155
156 class Year:
157     node_id = 0
158     number = 0
159
160     def __init__(self, node_id, number):
161         self.node_id = node_id
162         self.number = number
163         return
164
165
166 idc = -1
167
168
169 def put_paper(title, year_id):
170     global papers_array
171     global idc
172
173     title = title.upper()
174
175     for paper in papers_array:
176         if paper.title == title:
177             return paper.node_id
178
179     idc = idc + 1
```

```
180     paper_obj = Paper(idc, title, year_id)
181     papers_array.append(paper_obj)
182     return idc
183
184
185 def put_author(name):
186     global authors_array
187     global idc
188
189     for author in authors_array:
190         if author.name == name:
191             return author.node_id
192
193     idc = idc + 1
194     author_obj = Author(idc, name)
195     authors_array.append(author_obj)
196     return idc
197
198
199 def put_key_word(name):
200     global key_words_array
201     global idc
202
203     name = name.lower()
204
205     for key_word in key_words_array:
206         if key_word.name == name:
207             return key_word.node_id
208
209     idc = idc + 1
210     key_word_obj = Key_Word(idc, name)
211     key_words_array.append(key_word_obj)
212     return idc
```

```
213
214
215 def put_institution(name):
216     global isntitutions_array
217     global idc
218
219     name = name.upper()
220
221     for institution in institutions_array:
222         if institution.name == name:
223             return institution.node_id
224
225     idc = idc + 1
226     institution_obj = Institution(idc, name)
227     institutions_array.append(institution_obj)
228     return idc
229
230
231 def put_reference(name):
232     global references_array
233     global idc
234
235     name = name.upper()
236
237     for reference in references_array:
238         if reference.name == name:
239             return reference.node_id
240
241     idc = idc + 1
242     reference_obj = Reference(idc, name)
243     references_array.append(reference_obj)
244     return idc
245
```

```
246
247 def put_year(number):
248     global years_array
249     global idc
250
251     for year in years_array:
252         if year.number == number:
253             return year.node_id
254
255     idc = idc + 1
256     year_obj = Year(idc, number)
257     years_array.append(year_obj)
258     return idc
259
260
261 def write_ora_metanetwork_file(papers_array, authors_array,
        key_words_array, institutions_array, references_array,
        years_array):
262 #     papers_dict = {}
263 #     authors_dict = {}
264 #     key_words_dict = {}
265 #     institutions_dict = {}
266
267     for paper in papers_array:
268         paper.title = paper.title.replace('&', 'and')
269         paper.title = paper.title.replace('"', "'")
270
271     for author in authors_array:
272         author.name = author.name.replace('&', 'and')
273         author.name = author.name.replace('"', "'")
274
275     for key_word in key_words_array:
276         key_word.name = key_word.name.replace('&', 'and')
```

```
277             key_word.name = key_word.name.replace('"', "'")
278
279         for institution in institutions_array:
280             institution.name = institution.name.replace('&', 'and
                ')
281             institution.name = institution.name.replace('"', "'")
282
283         for reference in references_array:
284             reference.name = reference.name.replace('&', 'and')
285             reference.name = reference.name.replace('"', "'")
286             reference.name = reference.name.replace('<', '(')
287             reference.name = reference.name.replace('>', ')')
288
289         output = open('a.xml', 'w')
290
291 #         node_id = 1
292
293 #         node_ids_dict = {}
294
295 #         for paper in papers_dict:
296 #             node_ids_dict[paper] = node_id
297 #             node_id = node_id + 1
298
299 #         for author in authors_dict:
300 #             node_ids_dict[author] = node_id
301 #             node_id = node_id + 1
302
303 #         for key_word in key_words_dict:
304 #             node_ids_dict[key_word] = node_id
305 #             node_id = node_id + 1
306
307 #         for institution in institutions_dict:
308 #             node_ids_dict[institution] = node_id
```

```
309 #         node_id = node_id + 1
310
311     output.write('<?xml version="1.0" standalone="yes"?>' + '
            \n')
312     output.write('\n')
313     output.write('<DynamicMetaNetwork id="amn - 1">' + '\n')
314     output.write('<MetaNetwork id="amn - 1">' + '\n')
315
316     output.write('<nodes>' + '\n')
317
318     output.write('<nodeclass type="Event" id="Papers">' + '\n
            ')
319     for paper in papers_array:
320         output.write('<node id="' + str(paper.node_id) + '"
                title="' + paper.title + '"/>' + '\n')
321     output.write('</nodeclass>' + '\n')
322
323     output.write('<nodeclass type="Agent" id="Authors">' + '\
            n')
324     for author in authors_array:
325         output.write('<node id="' + str(author.node_id) + '"
                title="' + author.name + '"/>' + '\n')
326     output.write('</nodeclass>' + '\n')
327
328     output.write('<nodeclass type="Knowledge" id="Key Words">
            ' + '\n')
329     for key_word in key_words_array:
330         output.write('<node id="' + str(key_word.node_id) + '
                " title="' + key_word.name + '"/>' + '\n')
331     output.write('</nodeclass>' + '\n')
332
333     output.write('<nodeclass type="Organization" id="
            Institutions">' + '\n')
```

```
334     for institution in institutions_array:
335         output.write('<node id="' + str(institution.node_id)
              + '" title="' + institution.name + '"/>' + '\n')
336     output.write('</nodeclass>' + '\n')
337
338     output.write('<nodeclass type="Knowledge" id="References
          ">' + '\n')
339     for reference in references_array:
340         output.write('<node id="' + str(reference.node_id) +
              '" title="' + reference.name + '"/>' + '\n')
341     output.write('</nodeclass>' + '\n')
342
343     output.write('<nodeclass type="Event" id="Years">' + '\n'
          )
344     for year in years_array:
345         output.write('<node id="' + str(year.node_id) + '"
              title="' + str(year.number) + '"/>' + '\n')
346     output.write('</nodeclass>' + '\n')
347
348     output.write('</nodes>' + '\n')
349
350     output.write('<networks>' + '\n')
351
352     output.write('<network sourceType="Event" source="Papers"
          targetType="Agent" target="Authors" id="Papers x
          Authors" isDirected="true" allowSelfLoops="true">' + '
          \n')
353     for paper in papers_array:
354         for i in range(1, paper.authors_num + 1):
355                 output.write('<link source="' + str(paper.
                      node_id) + '" target="' + str(paper.
                      __dict__['author ' + str(i)][0]) + '"
                      value="' + str(paper.__dict__['author ' +
```

```
                    str(i)][1]) + '"/>' + '\n')
356     output.write('</network>' + '\n')
357
358     output.write('<network sourceType="Event" source="Papers"
            targetType="Knowledge" target="Key Words" id="Papers
            x Key Words" isDirected="true" allowSelfLoops="true">'
             + '\n')
359     for paper in papers_array:
360         for i in range(1, paper.key_words_num + 1):
361                 output.write('<link source="' + str(paper.
                        node_id) + '" target="' + str(paper.
                        __dict__['key_word ' + str(i)][0]) + '"
                        value="' + str(paper.__dict__['key_word '
                        + str(i)][1]) + '"/>' + '\n')
362     output.write('</network>' + '\n')
363
364     output.write('<network sourceType="Agent" source="Authors
            " targetType="Organization" target="Institutions" id="
            Authors x Institutions" isDirected="true"
            allowSelfLoops="true">' + '\n')
365     for author in authors_array:
366         for i in range(1, author.institutions_num + 1):
367                 output.write('<link source="' + str(author.
                        node_id) + '" target="' + str(author.
                        __dict__['institution ' + str(i)][0]) + '"
                        value="' + str(author.__dict__['
                        institution ' + str(i)][1]) + '"/>' + '\n
                        ')
368     output.write('</network>' + '\n')
369
370     output.write('<network sourceType="Event" source="Papers"
            targetType="Knowledge" target="References" id="Papers
            x References" isDirected="true" allowSelfLoops="true
```

```
        ">' + '\n')
371     for paper in papers_array:
372         for i in range(1, paper.references_num + 1):
373                 output.write('<link source="' + str(paper.
                        node_id) + '" target="' + str(paper.
                        __dict__['reference ' + str(i)][0]) + '"
                        value="' + str(paper.__dict__['reference '
                         + str(i)][1]) + '"/>' + '\n')
374     output.write('</network>' + '\n')
375
376     output.write('<network sourceType="Event" source="Papers"
          targetType="Event" target="Years" id="Papers x Years"
          isDirected="true" allowSelfLoops="true">' + '\n')
377     for paper in papers_array:
378         output.write('<link source="' + str(paper.node_id) +
              '" target="' + str(paper.year[0]) + '" value="1"/>
              ' + '\n')
379     output.write('</network>' + '\n')
380
381     output.write('</networks>' + '\n')
382
383     output.write('</MetaNetwork>' + '\n')
384     output.write('</DynamicMetaNetwork>' + '\n')
385
386
387 def make_ntx_graph(papers_array, authors_array,
        key_words_array, institutions_array, references_array,
        years_array):
388     G = ntx.Graph()
389     H = ntx.Graph()
390     I = ntx.Graph()
391     J = ntx.Graph()
392     K = ntx.Graph()
```

```
393
394     for paper in papers_array:
395         G.add_node(paper.node_id, category='Paper', title=
                paper.title, year=paper.year[0], authors_num=paper
                .authors_num, key_words_num=paper.key_words_num,
                references_num=paper.references_num)
396         H.add_node(paper.node_id, category='Paper', title=
                paper.title, year=paper.year[0], authors_num=paper
                .authors_num, key_words_num=paper.key_words_num,
                references_num=paper.references_num)
397         J.add_node(paper.node_id, category='Paper', title=
                paper.title, year=paper.year[0], authors_num=paper
                .authors_num, key_words_num=paper.key_words_num,
                references_num=paper.references_num)
398         K.add_node(paper.node_id, category='Paper', title=
                paper.title, year=paper.year[0], authors_num=paper
                .authors_num, key_words_num=paper.key_words_num,
                references_num=paper.references_num)
399
400     for author in authors_array:
401         G.add_node(author.node_id, category='Author', name=
                author.name, institutions_num=author.
                institutions_num)
402         I.add_node(author.node_id, category='Author', name=
                author.name, institutions_num=author.
                institutions_num)
403
404     for key_word in key_words_array:
405         H.add_node(key_word.node_id, category='Key word',
                name=key_word.name)
406
407     for institution in institutions_array:
408         I.add_node(institution.node_id, category='Institution
```

```
            ', name=institution.name)
409
410     for reference in references_array:
411         J.add_node(reference.node_id, category='Reference',
                name=reference.name)
412
413     for year in years_array:
414         K.add_node(year.node_id, category='Year', number=year
                .number)
415
416     for paper in papers_array:
417         for i in range(1, paper.authors_num + 1):
418             G.add_edge(paper.node_id, paper.__dict__['author␣
                    ' + str(i)][0], weight=paper.__dict__['author␣
                    ' + str(i)][1])
419
420     for paper in papers_array:
421         for i in range(1, paper.key_words_num + 1):
422             H.add_edge(paper.node_id, paper.__dict__['
                    key_word␣' + str(i)][0], weight=paper.__dict__
                    ['key_word␣' + str(i)][1])
423
424     for author in authors_array:
425         for i in range(1, author.institutions_num + 1):
426             I.add_edge(author.node_id, author.__dict__['
                    institution␣' + str(i)][0], weight=author.
                    __dict__['institution␣' + str(i)][1])
427
428     for paper in papers_array:
429         for i in range(1, paper.references_num + 1):
430             J.add_edge(paper.node_id, paper.__dict__['
                    reference␣' + str(i)][0], weight=paper.
                    __dict__['reference␣' + str(i)][1])
```

```
431
432     for paper in papers_array:
433         K.add_edge(paper.node_id, paper.year[0], weight=1)
434
435     return (G, H, I, J, K)
436
437
438 def papers_per_year(G):
439     a = []
440     for n,d in G.nodes_iter(data='True'):
441         if d['category'] == 'Year':
442             if d['number'] != 0:
443                 a.append((d['number'], G.degree(n)))
444     a.sort()
445     b = npy.array(a)
446     plt.xlabel
447     plt.bar(b[:,0], b[:,1], width=1.0, color='green')
448     plt.xticks(b[:,0], rotation=45.00)
449     plt.grid(True)
450     plt.xlabel('\n␣years')
451     plt.ylabel('papers')
452     plt.title('Papers␣per␣year␣\n')
453 #     plt.plot([1,2,3], [1,2,3], 'go-', label='line 1',
    linewidth=2)
454 #     plt.plot([1,2,3], [1,4,9], 'rs',  label='line 2')
455 #     plt.axis([0, 4, 0, 10])
456 #     plt.legend()
457     plt.show()
458
459
460 institution_in_form = True
461 gotofi0 = False
462
```

```
463 log = open('citesnake_log.txt', 'w')
464
465 log.write('Citesnake ran at ' + time.asctime() + '.\n\n')
466 log.write('The following problems emerged along the way:\n')
467
468 row_counter = -1
469 no_key_word_prov = 0
470 no_key_word_paps = 0
471 no_institution_prov = 0
472 no_institution_auts = 0
473
474
475 def make_wos_net():
476     global fi
477
478     global treshold
479
480     global wos_files_list
481
482     paper_reg_exp = wos.paper_reg_exp
483     author_reg_exp = wos.author_reg_exp
484     key_word_reg_exp = wos.key_word_reg_exp
485     institution_reg_exp = wos.institution_reg_exp
486     institution_reg_exp_fi0 = wos.institution_reg_exp_fi0
487     reference_reg_exp = wos.reference_reg_exp
488     year_reg_exp = wos.year_reg_exp
489
490     paper_col = wos.paper_col
491     author_col = wos.author_col
492     key_word_col = wos.key_word_col
493     institution_col = wos.institution_col
494     reference_col = wos.reference_col
495     year_col = wos.year_col
```

```
496
497     global papers_array
498     global authors_array
499     global key_words_array
500     global institutions_array
501     global references_array
502     global years_array
503
504     global idc
505
506     global institution_in_form
507     global gotofi0
508
509     global log
510
511     global row_counter
512     global no_key_word_prov
513     global no_key_word_paps
514     global no_institution_prov
515     global no_institution_auts
516
517     for f in wos_files_list:
518         if verbose:
519             print f
520
521         input_0 = csv.reader(open(wos_files_dir + f, 'r'),
                delimiter = '␣␣␣␣␣␣␣␣')
522         _line_0 = input_0.next()
523         for row in input_0:
524             find_all_obj = paper_reg_exp.findall(row[
                    paper_col])
525             year_f_all_obj = year_reg_exp.findall(row[
                    year_col])
```

```
526             year_int_num = 0
527             for year in year_f_all_obj:
528                 year_int_num = int(year)
529             for title in find_all_obj:
530                 paper_id = put_paper(title, put_year(
                        year_int_num))
531
532
533         input_1 = csv.reader(open(wos_files_dir + f, 'r'),
                delimiter = '␣␣␣␣␣␣␣␣')
534         _line_0 = input_1.next()
535         for row in input_1:
536             papers_fao = paper_reg_exp.findall(row[paper_col
                    ])
537             authors_fao = author_reg_exp.findall(row[
                    author_col])
538             for name in authors_fao:
539                 author_id = put_author(wos.author_std_form(
                        name))
540                 for title in papers_fao:
541                     title = title.upper()
542                     for paper in papers_array:
543                         if paper.title == title:
544                             paper.set_author(author_id)
545
546
547         input_2 = csv.reader(open(wos_files_dir + f, 'r'),
                delimiter = '␣␣␣␣␣␣␣␣')
548         _line_0 = input_2.next()
549         row_counter = -1
550         for row in input_2:
551             row_counter = row_counter + 1
552             papers_fao = paper_reg_exp.findall(row[paper_col
```

```
                ])
553             key_words_fao = key_word_reg_exp.findall(row[
                    key_word_col])
554             if len(row[key_word_col]) == 0:
555                 log.write('␣-␣Row␣' + str(row_counter) + '␣of
                        ␣file␣"' + f + '":␣no␣key␣word␣were␣
                        provided␣(' + str(len(papers_fao)) + '␣
                        paper(s)␣found␣in␣this␣row).\n')
556                 no_key_word_prov = no_key_word_prov + 1
557                 no_key_word_paps = no_key_word_paps + len(
                        papers_fao)
558             for name in key_words_fao:
559                 key_word_id = put_key_word(name)
560                 for title in papers_fao:
561                     title = title.upper()
562                     for paper in papers_array:
563                         if paper.title == title:
564                             paper.set_key_word(key_word_id)
565
566
567         input_3 = csv.reader(open(wos_files_dir + f, 'r'),
                delimiter = '␣␣␣␣␣␣␣␣')
568         _line_0 = input_3.next()
569         row_counter = -1
570         for row in input_3:
571             row_counter = row_counter + 1
572             institution_in_form = True
573             gotofi0 = False
574             authors_fao = author_reg_exp.findall(row[
                    author_col])
575             institutions_fao = institution_reg_exp.findall(
                    row[institution_col])
576             if len(row[institution_col]) == 0:
```

```
577                 log.write('␣-␣Row␣' + str(row_counter) + '␣of
                        ␣file␣"' + f + '":␣no␣institution␣name␣was
                        ␣provided␣(' + str(len(authors_fao)) + '␣
                        author(s)␣found␣in␣this␣row).\n')
578                 no_institution_prov = no_institution_prov + 1
579                 no_institution_auts = no_institution_auts +
                        len(authors_fao)
580
581             if len(institutions_fao) == 0 and fi == 2:
582                 institution_in_form = False
583                 institutions_fao = institution_reg_exp_fi0.
                        findall(row[institution_col])
584                 if len(institutions_fao) == 1:
585                     gotofi0 = True
586                 if len(institutions_fao) >= 2:
587                     row_institution_col = ""
588                     for i in range(0, len(authors_fao)):
589                         if i < len(institutions_fao):
590                             row_institution_col =
                                    row_institution_col + '[' +
                                    authors_fao[i] + ']␣' +
                                    institutions_fao[i] + ";␣"
591                         else:
592                             row_institution_col =
                                    row_institution_col + '[' +
                                    authors_fao[i] + ']␣' +
                                    institutions_fao[0] + ";␣"
593                     institutions_fao = institution_reg_exp.
                            findall(row_institution_col)
594                     institution_in_form = True
595
596             if len(institutions_fao) == 0 and fi == 1:
597                 institution_in_form = False
```

```
598                 institutions_fao = institution_reg_exp_fi0.
                        findall(row[institution_col])
599                 if len(institutions_fao) >= 1:
600                     row_institution_col = "["
601                     for i in range(0, len(authors_fao)-1):
602                         row_institution_col =
                                row_institution_col + authors_fao[
                                i] + "; "
603                     row_institution_col = row_institution_col
                            + authors_fao[len(authors_fao)-1] + '
                            ] ' + institutions_fao[0] + ";"
604                     institutions_fao = institution_reg_exp.
                            findall(row_institution_col)
605                     institution_in_form = True
606
607             if (len(institutions_fao) == 0 and fi == 0) or
                    gotofi0:
608                 institution_in_form = False
609                 institutions_fao = institution_reg_exp_fi0.
                        findall(row[institution_col])
610                 if len(institutions_fao) == 1:
611                     institution_name = institutions_fao[0]
612                     institution_authors_fao = authors_fao
613
614             for institution in institutions_fao:
615                 if institution_in_form:
616                     grouped_institution_reg_exp = wos.
                            grouped_institution_reg_exp
617                     institution_match_obj =
                            grouped_institution_reg_exp.match(
                            institution)
618                     institution_name = institution_match_obj.
                            group(wos.institution_nam_grp)
```

```
619                     institution_authors =
                            institution_match_obj.group(wos.
                            institution_aut_grp)
620                     institution_authors_reg_exp = wos.
                            institution_authors_reg_exp
621                     institution_authors_fao =
                            institution_authors_reg_exp.findall(
                            institution_authors)
622
623                 institution_id = put_institution(
                        institution_name)
624                 for name in institution_authors_fao:
625                     name_in_form = wos.author_std_form(name)
626                     for author in authors_array:
627                         if author.name == name_in_form:
628                             author.set_institution(
                                    institution_id)
629
630         input_4 = csv.reader(open(wos_files_dir + f, 'r'),
                delimiter = '        ')
631         _line_0 = input_4.next()
632         row_counter = -1
633         for row in input_4:
634             row_counter = row_counter + 1
635             papers_fao = paper_reg_exp.findall(row[paper_col
                    ])
636             references_fao = reference_reg_exp.findall(row[
                    reference_col])
637 #             if len(row[reference_col]) == 0:
638 #                 log.write(' - Row ' + str(row_counter) + '
    of file "' + f + '": no key word were provided (' + str(
    len(papers_fao)) + ' paper(s) found in this row).\n')
639 #                 no_key_word_prov = no_key_word_prov + 1
```

```
640 #                 no_key_word_paps = no_key_word_paps + len(
    papers_fao)
641             for name in references_fao:
642                 reference_id = put_reference(name)
643                 for title in papers_fao:
644                     title = title.upper()
645                     for paper in papers_array:
646                         if paper.title == title:
647                             paper.set_reference(reference_id)
648
649     log.write('␣-␣' + str(no_key_word_prov) + "␣rows␣didn't␣
        have␣any␣information␣on␣key␣word's␣column.\n")
650     pc0 = (float(no_key_word_paps) / float(len(papers_array))
        ) * 100
651     log.write('␣-␣(Until)␣' + str(no_key_word_paps) + '␣
        papers␣may␣not␣have␣any␣associated␣key␣word␣(~' + str(
        pc0) + '%).\n')
652
653     log.write('␣-␣' + str(no_institution_prov) + "␣rows␣didn'
        t␣have␣any␣information␣on␣institution's␣column.\n")
654     pc1 = (float(no_institution_auts) / float(len(
        authors_array))) * 100
655     log.write('␣-␣(Until)␣' + str(no_institution_auts) + '␣
        authors␣may␣not␣have␣an␣associated␣institution␣(~' +
        str(pc1) + '%).\n')
656
657     log.write('\n')
658     log.write('End␣of␣citesnake␣log␣file.\n')
659
660 #    write_ora_metanetwork_file(papers_array, authors_array,
    key_words_array, institutions_array)
661
662
```

```
663 def make_scp_net():
664     global fi
665 
666     global treshold
667 
668     global scp_files_list
669 
670     paper_reg_exp = scp.paper_reg_exp
671     author_reg_exp = scp.author_reg_exp
672     key_word_reg_exp = scp.key_word_reg_exp
673     institution_reg_exp = scp.institution_reg_exp
674     institution_reg_exp_fi0 = scp.institution_reg_exp_fi0
675     reference_reg_exp = scp.reference_reg_exp
676     year_reg_exp = scp.year_reg_exp
677 
678     paper_col = scp.paper_col
679     author_col = scp.author_col
680     key_word_col = scp.key_word_col
681     institution_col = scp.institution_col
682     reference_col = scp.reference_col
683     year_col = scp.year_col
684 
685     global papers_array
686     global authors_array
687     global key_words_array
688     global institutions_array
689     global references_array
690     global years_array
691 
692     global idc
693 
694     global institution_in_form
695     global gotofi0
```

```
696
697     global log
698
699     global row_counter
700     global no_key_word_prov
701     global no_key_word_paps
702     global no_institution_prov
703     global no_institution_auts
704
705     for f in scp_files_list:
706         if verbose:
707             print f
708
709         input_0 = csv.reader(open(scp_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
710         _line_0 = input_0.next()
711         for row in input_0:
712             find_all_obj = paper_reg_exp.findall(row[
                    paper_col])
713             year_f_all_obj = year_reg_exp.findall(row[
                    year_col])
714             year_int_num = 0
715             for year in year_f_all_obj:
716                 year_int_num = int(year)
717             for title in find_all_obj:
718                 paper_id = put_paper(title , put_year(
                        year_int_num))
719
720
721         input_1 = csv.reader(open(scp_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
722         _line_0 = input_1.next()
723         for row in input_1:
```

```
724             papers_fao = paper_reg_exp.findall(row[paper_col
                    ])
725             authors_fao = author_reg_exp.findall(row[
                    author_col])
726             for name in authors_fao:
727                 author_id = put_author(scp.author_std_form(
                        name))
728                 for title in papers_fao:
729                     title = title.upper()
730                     for paper in papers_array:
731                         if paper.title == title:
732                             paper.set_author(author_id)
733
734
735         input_2 = csv.reader(open(scp_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
736         _line_0 = input_2.next()
737         row_counter = -1
738         for row in input_2:
739             row_counter = row_counter + 1
740             papers_fao = paper_reg_exp.findall(row[paper_col
                    ])
741             key_words_fao = key_word_reg_exp.findall(row[
                    key_word_col])
742             if len(row[key_word_col]) == 0:
743                 log.write('␣-␣Row␣' + str(row_counter) + '␣of
                        ␣file␣"' + f + '":␣no␣key␣word␣were␣
                        provided␣(' + str(len(papers_fao)) + '␣
                        paper(s)␣found␣in␣this␣row).\n')
744                 no_key_word_prov = no_key_word_prov + 1
745                 no_key_word_paps = no_key_word_paps + len(
                        papers_fao)
746             for name in key_words_fao:
```

```
747                 key_word_id = put_key_word(name)
748                 for title in papers_fao:
749                     title = title.upper()
750                     for paper in papers_array:
751                         if paper.title == title:
752                             paper.set_key_word(key_word_id)
753
754
755         input_3 = csv.reader(open(scp_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
756         _line_0 = input_3.next()
757         row_counter = -1
758         for row in input_3:
759             row_counter = row_counter + 1
760             institution_in_form = True
761             gotofi0 = False
762             authors_fao = author_reg_exp.findall(row[
                    author_col])
763             institutions_fao = institution_reg_exp.findall(
                    row[institution_col])
764             if len(row[institution_col]) == 0:
765                 log.write('␣-␣Row␣' + str(row_counter) + '␣of
                        ␣file␣"' + f + '":␣no␣institution␣name␣was
                        ␣provided␣(' + str(len(authors_fao)) + '␣
                        author(s)␣found␣in␣this␣row).\n')
766                 no_institution_prov = no_institution_prov + 1
767                 no_institution_auts = no_institution_auts +
                        len(authors_fao)
768
769             if len(institutions_fao) == 0 and fi == 2:
770                 institution_in_form = False
771                 institutions_fao = institution_reg_exp_fi0.
                        findall(row[institution_col])
```

```
772                 if len(institutions_fao) == 1:
773                     gotofi0 = True
774                 if len(institutions_fao) >= 2:
775                     row_institution_col = ""
776                     for i in range(0, len(authors_fao)):
777                         if i < len(institutions_fao):
778                             row_institution_col =
                                row_institution_col + '[' +
                                authors_fao[i] + ']␣' +
                                institutions_fao[i] + ";␣"
779                         else:
780                             row_institution_col =
                                row_institution_col + '[' +
                                authors_fao[i] + ']␣' +
                                institutions_fao[0] + ";␣"
781                     institutions_fao = institution_reg_exp.
                        findall(row_institution_col)
782                     institution_in_form = True
783
784             if len(institutions_fao) == 0 and fi == 1:
785                 institution_in_form = False
786                 institutions_fao = institution_reg_exp_fi0.
                    findall(row[institution_col])
787                 if len(institutions_fao) >= 1:
788                     row_institution_col = "["
789                     for i in range(0, len(authors_fao)-1):
790                         row_institution_col =
                            row_institution_col + authors_fao[
                            i] + ";␣"
791                     row_institution_col = row_institution_col
                        + authors_fao[len(authors_fao)-1] + '
                        ]␣' + institutions_fao[0] + ";"
792                     institutions_fao = institution_reg_exp.
```

```
                    findall(row_institution_col)
793                 institution_in_form = True
794
795             if (len(institutions_fao) == 0 and fi == 0) or
                    gotofi0:
796                 institution_in_form = False
797                 institutions_fao = institution_reg_exp_fi0.
                        findall(row[institution_col])
798                 if len(institutions_fao) == 1:
799                     institution_name = institutions_fao[0]
800                     institution_authors_fao = authors_fao
801
802             for institution in institutions_fao:
803                 if institution_in_form:
804                     grouped_institution_reg_exp = scp.
                            grouped_institution_reg_exp
805                     institution_match_obj =
                            grouped_institution_reg_exp.match(
                            institution)
806                     institution_name = institution_match_obj.
                            group(scp.institution_nam_grp)
807                     institution_authors =
                            institution_match_obj.group(scp.
                            institution_aut_grp)
808                     institution_authors_reg_exp = scp.
                            institution_authors_reg_exp
809                     institution_authors_fao =
                            institution_authors_reg_exp.findall(
                            institution_authors)
810
811                 institution_id = put_institution(
                        institution_name)
812                 for name in institution_authors_fao:
```

```
813                     name_in_form = scp.author_std_form(name)
814                     for author in authors_array:
815                         if author.name == name_in_form:
816                             author.set_institution(
                                    institution_id)
817
818         input_4 = csv.reader(open(scp_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
819         _line_0 = input_4.next()
820         row_counter = -1
821         for row in input_4:
822             row_counter = row_counter + 1
823             papers_fao = paper_reg_exp.findall(row[paper_col
                    ])
824             references_fao = reference_reg_exp.findall(row[
                    reference_col])
825 #             if len(row[reference_col]) == 0:
826 #                 log.write(' - Row ' + str(row_counter) + '
    of file "' + f + '": no key word were provided (' + str(
    len(papers_fao)) + ' paper(s) found in this row).\n')
827 #                 no_key_word_prov = no_key_word_prov + 1
828 #                 no_key_word_paps = no_key_word_paps + len(
    papers_fao)
829             for name in references_fao:
830                 reference_id = put_reference(name)
831                 for title in papers_fao:
832                     title = title.upper()
833                     for paper in papers_array:
834                         if paper.title == title:
835                             paper.set_reference(reference_id)
836
837     log.write(' - ' + str(no_key_word_prov) + " rows didn't
            have any information on key word's column.\n")
```

```
838     pc0 = (float(no_key_word_paps) / float(len(papers_array))
            ) * 100
839     log.write('␣-␣(Until)␣' + str(no_key_word_paps) + '␣
            papers␣may␣not␣have␣any␣associated␣key␣word␣(~' + str(
            pc0) + '%).\n')
840
841     log.write('␣-␣' + str(no_institution_prov) + "␣rows␣didn'
            t␣have␣any␣information␣on␣institution's␣column.\n")
842     pc1 = (float(no_institution_auts) / float(len(
            authors_array))) * 100
843     log.write('␣-␣(Until)␣' + str(no_institution_auts) + '␣
            authors␣may␣not␣have␣an␣associated␣institution␣(~' +
            str(pc1) + '%).\n')
844
845     log.write('\n')
846     log.write('End␣of␣citesnake␣log␣file.\n')
847
848 #     write_ora_metanetwork_file(papers_array, authors_array,
        key_words_array, institutions_array)
849
850
851 def make_gsc_net():
852     global fi
853
854     global treshold
855
856     global gsc_files_list
857
858     paper_reg_exp = gsc.paper_reg_exp
859     author_reg_exp = gsc.author_reg_exp
860     year_reg_exp = gsc.year_reg_exp
861
862     paper_col = gsc.paper_col
```

```
863     author_col = gsc.author_col
864     year_col = gsc.year_col
865
866     global papers_array
867     global authors_array
868     global years_array
869
870     global idc
871
872     global log
873
874     global row_counter
875
876     for f in gsc_files_list:
877         if verbose:
878             print f
879
880         input_0 = csv.reader(open(gsc_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
881         _line_0 = input_0.next()
882         for row in input_0:
883             find_all_obj = paper_reg_exp.findall(row[
                    paper_col])
884             year_f_all_obj = year_reg_exp.findall(row[
                    year_col])
885             year_int_num = 0
886             for year in year_f_all_obj:
887                 year_int_num = int(year)
888             for title in find_all_obj:
889                 paper_id = put_paper(title, put_year(
                        year_int_num))
890
891
```

```
892             input_1 = csv.reader(open(gsc_files_dir + f, 'r'),
                    delimiter = ',', quotechar = '"')
893             _line_0 = input_1.next()
894             for row in input_1:
895                 papers_fao = paper_reg_exp.findall(row[paper_col
                        ])
896                 authors_fao = author_reg_exp.findall(row[
                        author_col])
897                 for name in authors_fao:
898                     author_id = put_author(gsc.author_std_form(
                            name))
899                     for title in papers_fao:
900                         title = title.upper()
901                         for paper in papers_array:
902                             if paper.title == title:
903                                 paper.set_author(author_id)
904
905     log.write('\n')
906     log.write('End of citesnake log file.\n')
907
908 #     write_ora_metanetwork_file(papers_array, authors_array,
        key_words_array, institutions_array)
909
910
911 def make_pmd_net():
912     global fi
913
914     global treshold
915
916     global pmd_files_list
917
918     paper_reg_exp = pmd.paper_reg_exp
919     paper_grp_reg = pmd.paper_grp_reg
```

```
920     author_reg_exp = pmd.author_reg_exp
921     year_reg_exp = pmd.year_reg_exp
922
923     paper_col = pmd.paper_col
924     author_col = pmd.author_col
925     year_col = pmd.year_col
926
927     global papers_array
928     global authors_array
929     global years_array
930
931     global idc
932
933     global log
934
935     global row_counter
936
937     for f in pmd_files_list:
938         if verbose:
939             print f
940
941         input_0 = csv.reader(open(pmd_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
942         _line_0 = input_0.next()
943         for row in input_0:
944             find_all_obj = paper_reg_exp.findall(row[
                    paper_col])
945             year_match_obj = year_reg_exp.match(row[year_col
                    ])
946             year_int_num = 0
947             if year_match_obj != None:
948                 year_int_num = int(year_match_obj.group(3))
949             for title in find_all_obj:
```

```
950                 title_mat_obj = paper_grp_reg.match(title)
951                 title_in_form = title_mat_obj.group(3)
952                 paper_id = put_paper(title_in_form, put_year(
                        year_int_num))
953
954
955         input_1 = csv.reader(open(pmd_files_dir + f, 'r'),
                delimiter = ',', quotechar = '"')
956         _line_0 = input_1.next()
957         for row in input_1:
958             papers_fao = paper_reg_exp.findall(row[paper_col
                    ])
959             authors_fao = author_reg_exp.findall(row[
                    author_col])
960             for name in authors_fao:
961                 author_id = put_author(pmd.author_std_form(
                        name))
962                 for title in papers_fao:
963                     title = title.upper()
964                     for paper in papers_array:
965                         if paper.title == title:
966                             paper.set_author(author_id)
967
968     log.write('\n')
969     log.write('End␣of␣citesnake␣log␣file.\n')
970
971 #     write_ora_metanetwork_file(papers_array, authors_array,
        key_words_array, institutions_array)
972
973
974 make_wos_net()
975 make_scp_net()
976 make_gsc_net()
```

```
977 make_pmd_net()
978 write_ora_metanetwork_file(papers_array, authors_array,
      key_words_array, institutions_array, references_array,
      years_array)
979 (G, H, I, J, K) = make_ntx_graph(papers_array, authors_array,
      key_words_array, institutions_array, references_array,
      years_array)
980 papers_per_year(K)
```

APÊNDICE B – CÓDIGO DO MÓDULO CSNAKE_WOS_MOD

Código do arquivo **csnake_wos_mod.py**.

```
1  #! /usr/bin/python
2  #
3  # Citesnake_WOS: module for make Citesnake (C) able to read
       files exported
4  # from Web of Science
5  # Copyright (C) 2011 Jose Sergio Bleckmann Reis Junior
6  #
7  # This work is licensed under the Creative Commons
8  # Attribution-NonCommercial-NoDerivs 3.0 Unported License. To
       view a copy of
9  # this license, visit http://creativecommons.org/licenses/by-
       nc-nd/3.0/ or
10 # send a letter to Creative Commons, 444 Castro Street, Suite
       900,
11 # Mountain View, California, 94041, USA.
12 #
13 # rev15 = regular expression version 1.5
14
15 import re
16
17 paper_reg_exp = re.compile("([A-Za-z0-9\x80-\xFF=~\,.;'!@#$
       %&*()_+[]{^|<>:␣-/?]+)")#rev15
18 author_reg_exp = re.compile("([A-Za-z0-9\x80-\xFF=~\,.'!@#$
       %&*()_+[]{^|<>:-][A-Za-z0-9\x80-\xFF=~\,.'!@#$%&*()_
       +[]{^|<>:␣-]+)")#rev15
19 key_word_reg_exp = re.compile("([^␣;][A-Za-z0-9\x80-\xFF
       =~\,.'!@#$%&*()_+[]{^|<>:␣-]+)")#rev15
20 institution_reg_exp = re.compile("([[][A-Za-z0-9\x80-\xFF
```

```
    =~\,.;'!@#$%&*()_+[]{^|<>:␣-]+[]␣][A-Za-z0-9=~\.;'!@#$
    %&*()_+[]{^|<>:␣-]+[,])")#rev15
21  reference_reg_exp = re.compile("([^␣;][A-Za-z0-9\x80-\xFF
    =~\,./'!@#$%&*()_+[]{^|<>:␣-]+)")#rev15
22  year_reg_exp = re.compile("[0-9]+")#rev15
23
24  paper_col = 8
25  author_col = 5
26  key_word_col = 19
27  institution_col = 22
28  reference_col = 29
29  year_col = 41
30
31  institution_reg_exp_fi0 = re.compile("([A-Za-z0-9=\x80-\xFF
    ~\,.'!@#$%&*()_+[]{^|<>:␣-/?]+)")#rev15
32  grouped_institution_reg_exp = re.compile("(([[]) ([A-Za-z0-9\
    x80-\xFF=~\,.;'!@#$%&*()_+[]{^|<>:␣-]+) ([]]) ([␣]) ([A-Za-z0
    -9\x80-\xFF=~\.;'!@#$%&*()_+[]{^|<>:␣-]+) ([ ,]))")#rev15
33  institution_nam_grp = 6
34  institution_aut_grp = 3
35  institution_authors_reg_exp = re.compile("([A-Za-z0-9\x80-\
    xFF=~\,.'!@#$%&*()_+[]{^|<>:-][A-Za-z0-9\x80-\xFF=~\,.'!@#
    $%&*()_+[]{^|<>:␣-]+)")#rev15
36
37  def extract_initils(first_names):
38      initials = ""
39      is_initial = False
40      for c in first_names:
41          if is_initial:
42              initials = initials + c + ".␣"
43              is_initial = False
44          if c == "␣": is_initial = True
45      return initials
```

```
46
47 def author_std_form(name):
48     name_reg_exp = re.compile("(([^,]+)([,])([A-Za-z0-9\x80-\
       xFF=~\.;'!@#$%&*()_+[]{^|<>:␣-]+))")#rev15
49     name_mat_obj = name_reg_exp.match(name)
50     name_initils = "␣No_initials"
51     name_lst_nam = "No_last_name"
52     if name_mat_obj != None: name_initils = name_mat_obj.
       group(4)
53     if name_mat_obj != None: name_lst_nam = name_mat_obj.
       group(2)
54     name_initils = extract_initils(name_initils)
55     name_in_form = name_initils + name_lst_nam
56     name_in_form = name_in_form.upper()
57     return name_in_form
```

APÊNDICE C – CÓDIGO DO MÓDULO CSNAKE_SCP_MOD

Código do arquivo **csnake_scp_mod.py**.

```
1  #! /usr/bin/python
2  #
3  # Citesnake_SCP: module for make Citesnake (C) able to read
       files exported
4  # from Scopus
5  # Copyright (C) 2011 Jose Sergio Bleckmann Reis Junior
6  #
7  # This work is licensed under the Creative Commons
8  # Attribution-NonCommercial-NoDerivs 3.0 Unported License. To
       view a copy of
9  # this license, visit http://creativecommons.org/licenses/by-
       nc-nd/3.0/ or
10 # send a letter to Creative Commons, 444 Castro Street, Suite
       900,
11 # Mountain View, California, 94041, USA.
12 #
13 # rev15 = regular expression version 1.5
14 # rev16:  \x22 added
15 # rev17:  \] added
16
17 import re
18
19 paper_reg_exp = re.compile("([A-Za-z0-9\x22\x80-\xFF=~\,.;'!@
       #$%&*()_+[}{^|<>:␣-/?]+)")#rev16
20 author_reg_exp = re.compile("([A-Za-z0-9\x22\x80-\xFF=~\.;'!@
       #$%&*()_+[}{^|<>:-][A-Za-z0-9\x80-\xFF=~\.;'!@#$%&*()_
       +[}{^|<>:␣-]+)")#rev16
21 key_word_reg_exp = re.compile("([^␣;][A-Za-z0-9\x22\x80-\xFF
```

```
    =~\,.'!@#$%&*()_+[}{^|<>:␣-]+)")#rev16
22  institution_reg_exp = re.compile("([A-Za-z0-9\x22\x80-\xFF
    =~\.;'!@#$%&*()_+[}{^|<>:␣-]+[,][A-Za-z0-9\x80-\xFF=~\.;'!
    @#$%&*()_+[}{^|<>:␣-]+[,][A-Za-z0-9\x80-\xFF=~\.,'!@#$
    %&*()_+[}{^|<>:␣-]+[;]*)")#rev15
23  reference_reg_exp = re.compile("([^␣;][A-Za-z0-9\x22\x80-\xFF
    =~\,./'!@#$%&*()_+[\]}{^|<>:␣-]+)")#rev15
24  year_reg_exp = re.compile("[0-9]+")#rev15
25
26  paper_col = 1
27  author_col = 0
28  key_word_col = 15
29  institution_col = 13
30  reference_col = 22
31  year_col = 2
32
33  institution_reg_exp_fi0 = re.compile("([A-Za-z0-9\x22\x80-\
    xFF=~\,.'!@#$%&*()_+[}{^|<>:␣-/?]+)")#rev15
34  grouped_institution_reg_exp = re.compile("(([A-Za-z0-9\x22\
    x80-\xFF=~\.;'!@#$%&*()_+[}{^|<>:␣-]+[,][A-Za-z0-9\x80-\
    xFF=~\.;'!@#$%&*()_+[}{^|<>:␣-]+)([,][␣])([A-Za-z0-9\x80-\
    xFF=~\.'!@#$%&*()_+[}{^|<>:␣-]+[,][A-Za-z0-9\x80-\xFF
    =~\.'!@#$%&*()_+[}{^|<>:␣-]*)([,]?)([A-Za-z0-9\x80-\xFF
    =~\.,'!@#$%&*()_+[}{^|<>:␣-]*)([;]*))")#rev15
35  institution_nam_grp = 4
36  institution_aut_grp = 2
37  institution_authors_reg_exp = re.compile("([A-Za-z0-9\x22\x80
    -\xFF=~\,.'!@#$%&*()_+[}{^|<>:-][A-Za-z0-9\x80-\xFF=~\,.'!
    @#$%&*()_+[}{^|<>:␣-]+)")#rev15
38
39  def author_std_form(name):
40      name_reg_exp = re.compile("(([^,.]+)([,]?)([␣])(([A-Za-z0
            -9\x22\x80-\xFF=~\;'!@#$%&*()_+[}{^|<>:␣-][.])+))")#
```

```
        rev15
41      name_mat_obj = name_reg_exp.match(name)
42      name_initils = "No_initials␣"
43      name_lst_nam = "No_last_name"
44      if name_mat_obj != None: name_initils = name_mat_obj.
            group(5)
45      if name_mat_obj != None: name_lst_nam = name_mat_obj.
            group(2)
46      name_initils = name_initils.replace(".", ".␣")
47      name_in_form = name_initils + name_lst_nam
48      name_in_form = name_in_form.upper()
49      return name_in_form
```

APÊNDICE D – CÓDIGO DO MÓDULO CSNAKE_GSC_MOD

Código do arquivo **csnake_gsc_mod.py**.

```
1  #! /usr/bin/python
2  #
3  # Citesnake_GSC: module for make Citesnake (C) able to read
       files exported
4  # from Google Scholar via Publish Or Perish (Harzing, 2008)
5  # Copyright (C) 2011 Jose Sergio Bleckmann Reis Junior
6  #
7  # This work is licensed under the Creative Commons
8  # Attribution-NonCommercial-NoDerivs 3.0 Unported License. To
       view a copy of
9  # this license, visit http://creativecommons.org/licenses/by-
       nc-nd/3.0/ or
10 # send a letter to Creative Commons, 444 Castro Street, Suite
       900,
11 # Mountain View, California, 94041, USA.
12 #
13 #  I'm guessing that the authors' initials are all letters.
14 #  rev10 = regular expression version 1.0
15
16 import re
17
18 paper_reg_exp = re.compile("([A-Za-z0-9\x80-\xFF=~\,.;'!@#$
       %&*()_+[]{^|<>:␣-/?]+)")#rev15
19 author_reg_exp = re.compile("([A-Za-z0-9\x80-\xE1\xE3-\xFF
       =~\;'!@#$%&*()_+[]{^|<>:-]+[␣][A-Za-z0-9\x80-\xE1\xE3-\xFF
       =~\;'!@#$%&*()_+[]{^|<>:␣-]+[\xE2]?)")#rev15
20 year_reg_exp = re.compile("[0-9]+")#rev15
21
```

```
22 paper_col = 2
23 author_col = 1
24 year_col = 3
25
26 def author_std_form(name):
27     name_reg_exp = re.compile("(([A-Za-z0-9\x80-\xE1\xE3-\xFF
       =~\;'!@#$%&*()_+[]{^|<>:-]+)([ ])([A-Za-z0-9\x80-\xE1\
       xE3-\xFF=~\;'!@#$%&*()_+[]{^|<>: -]+)([\xE2]?))")#
       rev15
28     name_mat_obj = name_reg_exp.match(name)
29     name_initils = "No_initials "
30     name_lst_nam = "No_last_name"
31     if name_mat_obj != None: name_initils = name_mat_obj.
       group(2)
32     if name_mat_obj != None: name_lst_nam = name_mat_obj.
       group(4)
33     name_ult_ini = ""
34     for char in name_initils:
35         name_ult_ini = name_ult_ini + char + ". "
36     name_in_form = name_ult_ini + name_lst_nam
37     name_in_form = name_in_form.upper()
38     return name_in_form
```

APÊNDICE E – CÓDIGO DO MÓDULO CSNAKE_PMD_MOD

Código do arquivo **csnake_pmd_mod.py**.

```
1  #! /usr/bin/python
2  #
3  # Citesnake_PMD: module for make Citesnake (C) able to read
       files exported
4  # from PubMeD
5  # Copyright (C) 2011 Jose Sergio Bleckmann Reis Junior
6  #
7  # This work is licensed under the Creative Commons
8  # Attribution-NonCommercial-NoDerivs 3.0 Unported License. To
       view a copy of
9  # this license, visit http://creativecommons.org/licenses/by-
       nc-nd/3.0/ or
10 # send a letter to Creative Commons, 444 Castro Street, Suite
       900,
11 # Mountain View, California, 94041, USA.
12 #
13 # rev15 = regular expression version 1.5
14
15 import re
16
17 paper_reg_exp = re.compile("([[]?[A-Za-z0-9\x80-\xFF=~\,.;'!@
       #$%&*()_+[}{^|<>:␣-/?]+[]]?[.]+)")#rev15
18 paper_grp_reg = re.compile("(([[]?)([A-Za-z0-9\x80-\xFF
       =~\,.;'!@#$%&*()_+[}{^|<>:␣-/?]+)([]]?)([.]+))")#rev15
19 author_reg_exp = re.compile("([␣]?[^,]+[␣][A-Za-z0-9\x80-\xFF
       =~\;'!@#$%&*()_+[}{^|<>:-]+)")#rev15
20 year_reg_exp = re.compile("(([A-Za-z0-9\x80-\xFF=~\,;'!@#$
       %&*()_+[}{^|<>:␣-/?]+[.][␣]+)(([0-9])+))")#rev15
```

```
21
22 paper_col = 0#
23 author_col = 2#
24 year_col = 4#
25
26 def author_std_form(name):
27     name_reg_exp = re.compile("(([␣]?)([^,]+)([␣])(([A-Za-z0
       -9\x80-\xFF=~\;'!@#$%&*()_+[}{^|<>:-])+))")#rev15
28     name_mat_obj = name_reg_exp.match(name)
29     name_initils = "No_initials␣"
30     name_lst_nam = "No_last_name"
31     if name_mat_obj != None: name_initils = name_mat_obj.
       group(5)
32     if name_mat_obj != None: name_lst_nam = name_mat_obj.
       group(3)
33     name_ult_ini = ""
34     for char in name_initils:
35         name_ult_ini = name_ult_ini + char + ".␣"
36     name_in_form = name_ult_ini + name_lst_nam
37     name_in_form = name_in_form.upper()
38     return name_in_form
```

REFERÊNCIAS BIBLIOGRÁFICAS

1 UNESCO Institute for Statistics, UIS Bulletin on Science and Technology Statistics, 2 (2005).

2 UNESCO, UNESCO Science Report 2010 - The Current Status of Science around the World, UNESCO Publishing, Paris, França (2010). ISBN: 978-92-3-104132-7.

3 De Bellis, N, Bibliometrics and citation analysis: from the Science citation index to cybermetrics, Scarecrow Press, Lanham, Estados Unidos da América (2009). ISBN: 0810867133.

4 Wasserman, S e Faust, K, Social Network Analysis: Methods and Applications, Cambridge University Press, New York, Estados Unidos da América (1994).

5 De Solla Price, DJ, “Network of Scientific Papers”, Science, 149, pp. 510-515 (1965).

6 Egghe, L e Rousseau, R, Introduction to Informetrics, Elsevier, Amsterdam, Holanda (1990).

7 Wouters, P e Leydesdorff, L, “Has Price dream come true - is scientometrics a hard science”, Scientometrics, 31 (2), pp. 193-222 (1994).

8 Melin, G e Persson, O, “Studying research collaboration using co-authorships”, Scientometrics, 36 (3), pp. 363-377 (1996).

9 Melin, G e Persson, O, “Hotel cosmopolitan: A bibliometric study of collaboration at some European universities”, Journal Of The American Society For Information Science, 49 (1), pp. 43-48 (1998).

10 Ding, Y, Foo, S e Chowdhury, G, “A bibliometric analysis of collaboration in the field of information retrieval”, International Information & Library Review, 30, pp. 367–376 (1999).

11 Kundra, R e Kretschmer, H, “A new model of scientific collaboration Part 2. Collaboration patterns in Indian medicine”, Scientometrics, 46 (3), pp. 519-528 (1999).

12 Persson, O e Beckmann, M, “Locating the network of interacting authors in scientific specialties”, Scientometrics, 33, pp. 351–366 (1995).

13 Usdiken, B e Pasadeos, Y, “organizational analysis in north-america and europe - A comparison of cocitation networks”, Organization Studies, 16 (3), pp. 503-526 (1995).

14 White, HD e McCain, KW, “Visualization of literatures”, Annual Review Of Information Science And Technology, 32, pp. 99-168 (1997).

15 Kleinberg, JM, “Authoritative sources in a hyperlinked environment”, Journal Of The Association for Computing Machinery, 46 (5), pp. 604-632 (1999).

16 Chen, CM, “Visualising semantic spaces and author co-citation networks in digital libraries”, Information Processing & Management, 35 (3), pp. 401-420 (1999).

17 “Web of Knowledge”, http://apps.webofknowledge.com/ (2013).

18 “Scopus (Elsevier)”, http://www.scopus.com/scopus/home.url (2013).

19 “PubMed – NCBI”, http://www.ncbi.nlm.nih.gov/pubmed/ (2013).

20 “arXiv.org e-Print archive”, http://arxiv.org/ (2013).

21 “Google Scholar”, http://scholar.google.com/ (2013).

22 “CiteSeer - Pennsylvania State University”, http://citeseer.ist.psu.edu/ (2013).

23 “Wikipedia”, http://www.wikipedia.org/ (2013).

24 Huisman M e van Duijn MA, “Software for Social Network Analysis”, Models and Methods in Social Network Analysis, Editado por Carrington PJ, Scott J, e Wasserman S, pp. 270-316, Cambridge University Press, New York, Estados Unidos da América (2004).

25 “iOpener Project”, http://tangra.si.umich.edu/clair/iopener/ (2013).

26 “Science of Science Cyberinfrastructure”, http://sci.slis.indiana.edu/ (2013).

27 Garfield, E e Welljams-Dorof, A, “Citation data: their use as quantitative indicators for science and technology evaluation and policy-making”, Science & Public Policy, 19 (5), pp.321-327 (1992).

28 Cullen, SE, “Importance of citation analysis to IP portfolio management”, Intellectual Asset Management, 2008 (June/July), p. 7 (2008).

29 “QualiPSo”, http://www.qualipso.org (2013).

30 Delamaro, M, Maldonado, J e Jino, M, Introdução ao Teste de Software, Campus (2007).

31 Bernardo, PC e Kon, F, “A Importância dos Testes Automatizados - Controle ágil, rápido e confiável de qualidade”, Engenharia de Software Magazine, 1(3), pp. 54-57 (2008).

32 Beck, K, Test-Driven Development: By Example, Addison-Wesley Professional (2002).

33 Schwaber, K e Beedle, M, Agile Software Development with SCRUM, Prentice Hall (2001).

34 “Network Workbench”, http://http://nwb.cns.iu.edu/ (2013).

35 Reis Junior, JSB, Barroso, ACO, Imakuma, K e Menezes, MO, “News and Its Influence on The Viability of Nuclear Power Plants Deployment - A Modified Epidemiological Model for News Generation”, 2007 International Nuclear Atlantic Conference - INAC 2007, Santos, Brasil, 30 de Setembro-5 de Outubro (2007).

36 Reis Junior, JSB, Barroso, ACO e Menezes, MO, “Modeling news dissemination on nuclear issues”, 2011 International Nuclear Atlantic Conference - INAC 2011, Belo Horizonte, Brasil, 24-28 de Outubro (2011).

37 Newman, MEJ, “The structure of scientific collaboration networks”, Proceedings Of The National Academy Of Sciences Of The United States Of America, 98 (2), pp. 404-409 (2001).

38 Newman, MEJ, “Scientific collaboration networks. I. Network construction and fundamental results”, Physical Review E, 64 (1) (2001).

39 Newman, MEJ, “Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality”, Physical Review E, 64 (1) (2001).

40 Barabasi, AL, Jeong, H, Neda, Z et al., “Evolution of the social network of scientific collaborations”, Physica A-Statistical Mechanics And Its Applications, 311 (3-4), pp. 590-614 (2002).

41 Albert, R e Barabasi, AL, “Statistical mechanics of complex networks”, Reviews Of Modern Physics, 74 (1), pp. 47-97 (2002).

42 Dorogovtsev, SN e Mendes, JFF, “Evolution of networks”, Advances In Physics, 51 (4), pp. 1079-1187 (2002).

43 Barrat, A, Barthelemy, M, Pastor-Satorras, R et al., “The architecture of complex weighted networks”, Proceedings Of The National Academy Of Sciences Of The United States Of America, 101 (11), pp. 3747-3752 (2004).

44 Chen, C, “Searching for intellectual turning points: Progressive Knowledge Domain Visualization”, Proceedings of the National Academy of Sciences of the United States of America (PNAS), 101 (1), pp. 5303-5310 (2004).

45 Chen, CM, “CiteSpace II: Detecting and visualizing emerging trends and transient patterns in scientific literature”, Journal Of The American Society For Information Science And Technology, 57 (3), pp. 359-377 (2006).

46 Chen, C et al., “The structure and dynamics of co-citation clusters: A multiple-perspective co-citation analysis”, Journal of the American Society for Information Science and Technology (2010).

47 “CiteSpace: visualizing patterns and trends in scientific literature”, http://cluster.cis.drexel.edu/ cchen/citespace/ (2013).

48 Sipser, M, Introduction to the Theory of Computation, 2, Thomson Course Technology, Boston, Estados Unidos da América (2006).

49 “re — Regular expression operations”, http://docs.python.org/2.7/library/re.html (2013).

50 Harzing, AW, Publish or Perish, disponível em http://www.harzing.com/pop.htm (2007).

51 “*ORA”, http://www.casos.cs.cmu.edu/projects/ora/ (2013).

52 “NetworkX”, http://networkx.github.io/ (2013).

53 “Citesnake”, http://www.citesnake.mat.br/ (2014).

54 “wxPython”, http://www.wxPython.org/ (2014).

55 “Cyberinfrastructure Shell”, http://cishell.org/ (2014).

56 “Network Workbench”, http://nwb.cns.iu.edu/ (2014).

57 Reminga, J e Carley, KM, “Measures for ORA”, http://www.casos.cs.cmu.edu/publications/papers/reminga_2003_ora.pdf (2003).

58 The Generation IV International Forum, “Generation IV Systems”, https://www.gen-4.org/gif/jcms/c_40465/generation-iv-systems (2014).

59 The Generation IV International Forum, “SFR”, https://www.gen-4.org/gif/jcms/c_9361/sfr (2014).

60 The Generation IV International Forum, “SCWR”, https://www.gen-4.org/gif/jcms/c_9360/scwr (2014).

61 The Generation IV International Forum, “Technology Roadmap”, https://www.gen-4.org/gif/jcms/c_9352/technology-roadmap (2014).

62 Leydesdorff, L, “Software and data of Loet Leydesdorff”, http://www.leydesdorff.net/software.htm (2014).