# Uma Arquitetura para Detecção de Plágio Baseada em Redes P2P

Juan Andrés Mussini[1], Lau Cheuk Lung[2],
Fábio Favarim[3], Altair Olivo Santin[1]

[1]Programa de Pós-Graduação em Informática (PPGIA)
Pontifícia Universidade Católica do Paraná (PUCPR)
Curitiba–PR, Brasil, 80215-901

[2]Departamento de Informatica e Estatística (INE)
[3]Departamento de Automação e Sistemas (DAS)
Universidade Federal de Santa Catarina (UFSC)
Florianópolis–SC, Brasil, 88040-900

juan.mussini@pucpr.br,lau.lung@inf.ufsc.br
fabio@das.ufsc.br,santin@ppgia.pucpr.br

**Resumo** Hoje em dia a Internet se tornou uma referência em aquisição de informações. Mas isto pode ser mau utilizado, já que um usuário pode acessar uma informação e usá-la como de sua própria autoria. Isto caracteriza uma ato de plágio. Tem sido cada vez mais comum este tipo de ocorrência, e ferramentas para prevenção deste ato são necessárias. A fim de ajudar a lidar com este problema, é proprosto o PeerDetect, um sistema de detecção de plágio elaborado sobre uma infraestrutura P2P. A solução proposta permite ao usuário detectar plágios em nós que pertencem a uma rede P2P.

## 1 Introdução

O advento e a expansão da Internet trouxeram consideráveis e inquestionáveis benefícios para a socialização da informação. A Internet tem contribuído para uma disponibilização eficiente do conhecimento humano e sua aquisição. No entanto, essa socialização também trouxe um importante problema: o plágio. autoria. 2. Imitar trabalho alheio". No meio acadêmico em geral, e até mesmo no meio empresarial, tem surgido diversos casos de processos de plágios contra indivíduos ou empresas que se apoderam de conteúdos obtidos na Internet, assumindo a autoria da informação de forma fraudulenta. Na academia, por exemplo, em praticamente todas as áreas do conhecimento e níveis, professores e pesquisadores tem encontrado sérias dificuldades para determinar a originalidade de trabalhos acadêmicos e científicos de seus alunos e pares. Na grande maioria das vezes, o que pode ser grave, a determinação da originalidade de um trabalho tem sido relegada para um segundo plano, confiando na honestidade de pessoas e empresas. O plágio é um problema muito sério, pois consiste da apropriação indevida do trabalho intelectual de terceiros em beneficio do plagiador para este alcançar alguma vantagem financeira, ascensão social ou profissional. Isto faz do plágio um tópico controverso e muito comum atualmente.

XXXIV Conferencia Latinoamericana de Informática (CLEI 2008), pp. 1385-1394, 2008.

Ocorrências de plágios têm aumentado nos últimos anos [1], e ferramentas para fazer cópias cresceram na mesma proporção, desde uma simples busca em um indexador de conteúdo, como o Google [2], a sites que oferecem trabalhos já prontos [3,4]. Muitos professores, especialmente aqueles que não são de áreas relacionadas à computação, não possuem informação sobre técnicas de detecção deste tipo de fraudes. Comumente, para detecção de plágio em um determinado documento é utilizada a checagem manual, onde um professor seleciona frases aleatórias do documento e as utiliza para procurar por documentos iguais em uma ferramenta de busca. No entanto, esta checagem manual consome muito tempo e é um processo laborioso, fazendo com que não seja amplamente praticado.

Diante disso, algumas poucas ferramentas começaram a surgir recentemente [5,6] a fim de automatizar este processo. Estas ferramentas geralmente indicam qual o grau de similaridade entre dois documentos. Caso este grau seja considerado alto, uma checagem manual por um especialista, como por exemplo, um professor, é necessária. Esta checagem manual é essencial, já que não existe um sistema de detecção que garanta uma certeza de 100%.

Neste artigo é proposto um sistema de detecção de plágio em documentos escritos em linguagem natural chamado PeerDetect, o qual se utiliza de redes P2P. A abordagem apresentada permite criar comunidades de peers de acordo com uma determinada área e linha de pesquisa, tópico e assunto. Cada peer desta comunidade deve dispor seus documentos em um repositório a ser compartilhado na rede P2P. Uma vez configurada uma comunidade, os peers pertencentes a esta podem acessá-las para uma busca eficiente de documentos para detectar possíveis fraudes. Inicialmente, o algoritmo de detecção de plágio extrai informações sobre o documento analisado. Esta informação é comparada a outros documentos na rede P2P com o mesmo tópico de interesse. Documentos que possuem informações similares são transferidos para o nó que iniciou a busca e, após a transferência, um segundo e mais eficiente algoritmo de detecção de plágio é aplicado. Este algoritmo é executado localmente nesse nó. Os resultados deste segundo passo são apresentados para o usuário, para que seja feita uma avaliação mais profunda.

## 2 Arquiteturas P2P

Os sistemas Par-a-Par, conhecidos também por *Peer-to-Peer* (P2P), fornecem recursos (serviços, arquivos, etc) de forma descentralizada, onde cada nó pode tanto buscar por recursos como também provê-los. Diferente de sistemas convencionais, como os baseados no modelo cliente-servidor, onde os serviços (recursos) são de uma maneira usual concentrados no servidor, os sistemas P2P possibilitam a colaboração e acesso entre clientes e recursos de uma forma dispersa, sem a necessidade de um servidor central em um sistema distribuído. Uma das propriedades dos sistemas P2P é prover aos seus participantes, chamados de *peers* ou nós, um comportamento igualitário, isto é, com as mesmas capacidades e responsabilidades.

### 2.1 JXTA

Inicialmente, quando alguns protocolos e aplicações P2P começaram a ser utilizados, eles não interoperavam entre si. Com a intenção de resolver este problema, a Sun Microsystems anunciou a plataforma JXTA [7], que foi concebida a fim de padronizar um conjunto comum de protocolos abertos permitindo que qualquer dispositivo na rede se comunique e colabore na maneira das redes P2P. Os protocolos JXTA são definidos como um conjunto de mensagens XML.

Os protocolos JXTA estabelecem uma rede superposta virtual por sobre a Internet, permitindo que nós interajam diretamente e se auto-organizem independentemente da topologia e conectividade de sua rede. Desta maneira, muitas redes virtuais AD-HOC podem ser criadas e mapeadas dinamicamente a apenas uma rede física. JXTA define um conjunto mínimo de requisitos para que os nós formem e ingressem a redes P2P virtuais. Sendo assim, é possível que desenvolvedores de aplicações definam a topologia de rede que mais convém aos requisitos das suas aplicações. JXTA foi desenhado para ser independente de linguagem de programação, de sistema operacional e de protocolos de rede.

### 2.2 Localização de Recursos em Redes P2P

Um dos serviços mais importantes nas redes P2P é a localização de recursos. Este serviço permite aos usuários encontrem na rede P2P os recursos (serviço, arquivos, etc.) que necessitam. De forma geral, estes resultados consistem em uma lista de nós onde o recurso pode ser encontrado. Depois da localização do recurso, o nó solicitante pode acessar diretamente o seu dono. Basicamente há três abordagens para a localização de recursos em redes P2P: centralizada, por inundação e baseada em Tabelas *Hash* Distribuídas.

Na primeira abordagem, um único servidor mantém um catálogo dos dados que possui. Quando um nó ingressa no sistema, ele contata o servidor central e este lhe envia uma lista dos recursos disponíveis para os outros nós. Para localizar um recurso, um nó cliente envia sua busca ao nó central, o qual efetua uma busca em sua base de dados e responde com uma lista de nós que possuem o recurso desejado. Depois dessa consultar, o nó passa a se comunicar diretamente com os nós possuidores do recurso.

Na abordagem por inundação, todos os nôs são autônomos e têm o mesmo papel. Não há nó central e cada nó é responsável por manter um índice dos recursos que possui. Como os nós participantes são organizados de forma não-estruturada, cada nó pode se comunicar diretamente com alguns nós e indiretamente com todos os nós na rede. Assim, um nó pode não estar ciente de quais os outros nós na rede. Desta forma, para se executar a busca, uma mensagem com escopo limitado é enviada à rede. Esta funciona da seguinte maneira: cada requisição é primeiramente enviada aos nós conectados diretamente, que por sua vez enviam aos nós conectados diretamente a eles, e assim por diante até que o escopo seja alcançado (normalmente 5 a 9 passos). Como as mensagens possuem escopo limitado, é possível que nós não localizem os recursos mesmo que estes estejam disponíveis na rede.

**Distributed Hash Tables.** *Distributed Hash Tables* (DHT) são uma classe de sistemas distribuídos descentralizados que executam as funções de uma tabela *hash*. Esta tabela *hash* armazena duplas da forma {chave, valor} e os valores são procurados através da chave. Na DHT, as chaves são distribuídas entre os vários nós. Desta maneira, tanto o armazenamento quanto a busca são distribuídos entre os nós participantes. As DHT são projetadas para suportar constantes ingressos, saídas e falhas dos nós. Isto permite que suportem um grande número de nós. Ao contrário dos mecanismos de busca P2P convencionais (centralizado e por inundação), a DHT utiliza um mecanismo de busca baseado em chaves mais estruturado, a fim de obter tanto a descentralização do Gnutella quanto a eficiência de resultados do Napster. No entanto, a DHT tem uma desvantagem, pois somente suporta buscas com resultado exato (utilizando a chave), ao invés de busca por palavras-chave. Esta funcionalidade pode ser feita em uma camada acima da DHT [8].

## 3 Detecção de Plágio

A detecção de plágio pode ser dividida em dois tipos: detecção de plágio em códigos fontes e em documentos escritos em linguagens naturais. Para detecção em códigos fontes, as técnicas de contagem de atributos e de métrica de estruturas são as mais utilizadas [9]. Por ser mais simples, há mais literatura referente a plágio em códigos fontes do que em documentos escritos em linguagens naturais. Isso se deve a linguagem natural ser muito mais complexa e ambígua. Neste artigo será abordado somente plágio em documentos escritos em linguagem natural.

### 3.1 Plágio em Documentos Escritos em Linguagem Natural

A identificação de plágio em textos escritos em linguagem natural cabe normalmente ao professor ou tutor. Se eles estiveram familiarizados com o estilo de escrever do aluno, eles podem ser capazes de identificar irregularidades no trabalho do aluno se comparada a outros trabalhos do mesmo só que mais antigo, ou até mesmo identificar vocabulários e linguagens diferentes utilizados. Inicialmente, estas características podem identificar um plágio em potencial. Outras características suspeitas de plágios em documentos escritos em linguagem natural são:

- Uso de vocabulário: comparação de vocabulário com vocabulário conhecido. Quanto maior a diferença, ou seja, quanto mais palavras novas o documento tiver, menor a probabilidade de cópia;
- Mudança de vocabulário: se o vocabulário utilizado muda constantemente dentro de um mesmo texto, isto pode indicar um caso de cópia e cola;
- Pontuação: a pontuação varia muito de texto para texto, se for similar em dois documentos, pode ser um caso de cópia;
- Quantidade de similaridade entre textos - quanto maior a similaridade de termos comuns como nomes, definições, maior a suspeita;

– Erros de gramática comuns;

Há algoritmos mais simples que tentam detectar plágio utilizando poucas mas importantes características. Estas podem ser extraídas e comparadas em outro momento – ambos os documentos não necessitam ser comparados ao mesmo tempo. Um exemplo disto pode ser visto em [10], onde o algortimo utilizado consiste em encontrar palavras que aparecerem somente uma vez em todo o texto. Estas palavras são chamadas de *hapax legomena*. Documentos com *hapax legomena* similares podem ser uma cópia.

Outros algoritmos exploram o documento em um nível maior. Estes analizam mais informações, como estrutura do documento, extração de palavras-chaves (substantivos, verbos e adjetivos) e caracteríticas da estrutura, e necessitem de pelo menos dois documentos presentes para compará-los. Documentos com características similares podem ser uma cópia. SIM [11] e YAP3 [12] tem estas características.

## 4 PeerDetect

Nesta seção, é apresentado um sistema de detecção de plágio em documentos escritos em linguagem natural chamado **PeerDetect**. Este sistema é baseado no sistema Web2Peer [13]. O Web2Peer é uma infra-estrutura para publicação e recuperação de páginas *web* em uma rede P2P.

O PeerDetect possibilita a utilização, como fonte de dados, computadores pessoais conectados através de uma rede P2P. Todos os usuários na rede são nós que podem publicar documentos e também podem utilizar esta rede como uma fonte de dados para procurar por possíveis plágios nos documentos disponíveis em outros nós. PeerDetect utiliza as vantagens de redes P2P através protocolos JXTA para publicar e procurar documentos similares.

O JXTA possui seu próprio protocolo para a pesquisa de recursos, neste caso, documentos escritos em linguagens naturais. Porém, não é utilizado, visto que a busca é baseada em uma abordagem por inundação. Esta abordagem resulta em um tempo de resposta alto e em certas ocasiões, um recurso disponível na rede pode não ser encontrado devido ao escopo limitado da busca. Para superar estas características, foi utilizado um serviço de indexação baseado em tabelas *hash* distribuídas (DHT), a qual provê respostas mais rápidas e determinísticas, se comparadas com o mecanismo JXTA. A DHT cria uma relação entre a localização do documento, dentro da rede P2P, e uma ou mais palavras-chave. A abordagem apresentada do PeerDetect possui duas funcionalidades principais – publicar documentos disponíveis nos nós e procurar por plágios de um documento.

### 4.1 Publicação de Documentos

A publicação de documentos é um processo importante, onde se permite que outros nós tomem conhecimento da existência de um documento. Para publicar um documento, é necessário definir palavras-chave de acordo com o seu assunto, onde cada uma delas será utilizada para indexá-lo na DHT. Elas também serão

utilizadas no processo de verificação para que seja descoberto na DHT quais os documentos relacionados a estas palavras-chave. Estas palavras que possuem relação com o documento, são determinadas por um algoritmo específico (algoritmo inicial). Este algoritmo tem como característica a rapidez em extrair informações simples que diferem um documento do outro.

Para evitar o problema de haver comparações entre documentos de contextos diferentes (por exemplo, um documento sobre Computação@SistemasDistribuídos comparado com um da Medicina@Patologia) o PeerDetect define o conceito de comunidade de nós. A comunidade de nós é formada por nós que possuem documentos dentro de uma área de interesse específica. Sendo assim, ao publicar o documento, o usuário necessita especificar uma série de atributos, ou seja, um conjunto de palavras-chave, que define a comunidade de nós do documento. Estes atributos são: área de conhecimento do documento, linha de pesquisa, tópico e assunto. A Figura 1 apresenta a arquitetura do PeerDetect, onde é mostrado o Peer 0 passo-a-passo no processo de publicação e na seqüência, cada passo é explicado.

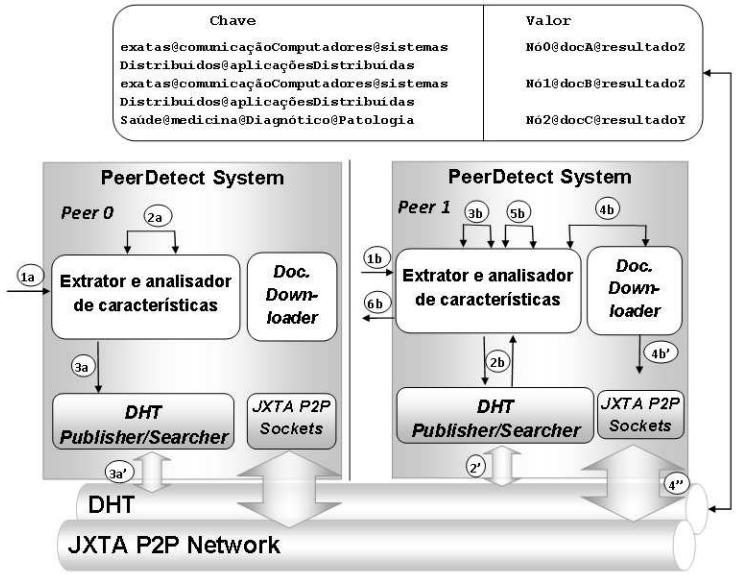



**Figura 1.** Processo de publicação (Peer 0) e de verificação (Peer 1)

1. a) O usuário define a comunidade de nós ao especificar a área do documento, linha de pesquisa, tópico e assunto. Mais adiante, o usuário necessita indicar qual o algoritmo inicial que será usado no passo seguinte;
2. a) O módulo "Extrator de características" aplica o algoritmo inicial no documento e uma lista de palavras-chave relacionada a ele é gerada ($docKeywords$);

3. a) O par $\langle chave, valor \rangle$ é inserido na DHT. A *chave* é composta pela concatenação dos atributos que definem a comunidade de nós (*area*, *linha*, *topico* e *assunto*). O *valor* é composto pela concatenação do identificador do nó ($peerId$), nome do documento ($docId$), identificador do algoritmo inicial ($algId$), e a lista de palavras-chave ($docKeywords$). A partir deste passo, o documento é disponibilizado na rede P2P, e pode ser encontrado pelos outros nós utilizando o mecanismo de busca disponibilizado pela DHT.

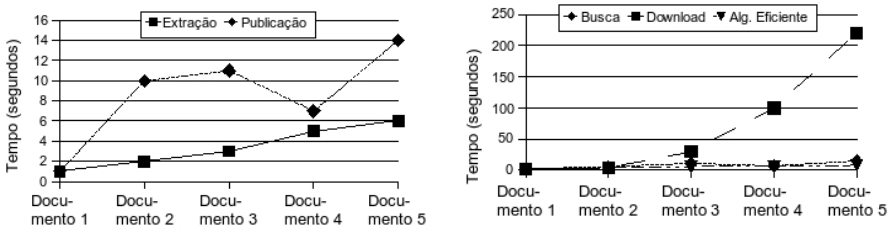
### 4.2 Verificação de Documentos

A verificação de documentos permite a um usuário verificar se um documento é plágio. O Peer 1 da Figura 1 apresenta passo a passo o processo de verificação e na seqüência, cada passo é explanado.

1. b) O usuário define a comunidade de nós que quer realizar a busca por documentos plagiados. Ele realiza este procedimento especificando a área, linha de pesquisa, tópico e assunto do documento. O usuário também deve indicar qual é o algoritmo inicial que será utilizado no passo 3b;
2. b) O PeerDetect utiliza os atributos dos documentos para formar a chave $area@linha@topico@assunto$ e executa uma busca na DHT utilizando esta chave. Esta busca é feita utilizando a interface "Publisher/Searcher" da DHT. Ela resulta em uma lista ($docList$) indicando todos os documentos na mesma comunidade de nós. Cada item nesta lista possui a seguinte informação: o identificador do nó $peerId$, o nome do documento $docId$, identificador do algoritmo inicial ($algId$) e a lista de palavras-chave ($docKeywords$) relacionadas ao documento;
3. b) O módulo "Extrator de características" aplica o algoritmo inicial ao documento sendo verificado e gera uma lista de palavras-chave sobre o documento ($docCheckKeywords$). O PeerDetect utiliza estas palavras-chave para determinar quais documentos da $docList$ são similares. Este processo consiste basicamente em comparar as $docCheckKeywords$ com as $docKeywords$ para cada item na $docList$. Todos os documentos similares são colocados em outra lista ($docList2$);
4. b) O módulo "Document Downloader" faz com que o PeerDetect receba todos os documentos indicados na $docList2$, utilizando a rede JXTA.
5. b) O módulo "Analisador de características" executa uma comparação de todos os documentos recebidos, utilizando um algoritmo mais eficiente a fim de determinar o grau de similaridade do documento sendo verificado. Uma lista ($plagiarismList$) com os documentos mais similares é gerada.
6. b) Finalmente, o PeerDetect informa ao usuário a $plagiarismList$. Posteriormente, o usuário pode executar uma verificação manual dos documentos na lista para certificar-se da veracidade dos resultados.

## 5 Avaliação

Uma boa ferramenta de detecção de plágio deve ser, além de tudo, útil, o que significa que a ferramenta deve ao menos apontar para a direção correta, para




(a) Publicação (b) Verificação

**Figura 2.** Tempo de publicação/verificação

que um especialista obtenha a menor quantidade de falsos positivos (indicação de plágio quando na verdade não é). É preciso também apresentar os resultados em um tempo aceitável. Em outras palavras, a ferramenta deve funcionar de forma correta em termos de acertos e eficiente no que se diz respeito ao tempo. Os experimentos foram realizados em um ambiente consistindo de 3 computadores, um AMD Turion 1.6 GHz com 1 GB ram (nó 0) e dois AMD Athlon XP 2.6 (nós 1 e 2) conectados a um rede local de 100Mb/s. O ambiente de *software* instalado nas máquinas foi o S.O. Linux 2.6.23 e máquina virtual Java de 32 bits versão 1.6.0.03.

A primeira avaliação foi o **tempo de publicação**, o que consiste no tempo de extração do algoritmo inicial (passo 2a da Figura 1) e o seu tempo de publicação (passo 3a da Figura 1). Os testes foram realizados com a utilização de documentos em formato PDF com diferentes números de páginas. Esses documentos possuem características apresentadas na Tabela 1. O algoritmo inicial utilizado foi o algoritmo apresentado em [10]. Este algoritmo detecta as chamadas *hapax legomena*. Os resultados são apresentados na Figura 2(a).

**Tabela 1.** Documentos utilizados para testes

| | Páginas | *hapax legomena* | Tamanho |
|---|---|---|---|
| Documento 1 | 1 | 77 | 38KB |
| Documento 2 | 6 | 655 | 83KB |
| Documento 3 | 13 | 944 | 450KB |
| Documento 4 | 90 | 473 | 733KB |
| Documento 5 | 93 | 1590 | 4.7MB |

Pode-se visualizar que o tempo de extração é relativo ao tamanho do documento, porém, isso não está diretamente relacionado ao tempo de publicação, visto que este tempo depende da quantidade de *hapax legomena* encontradas no documento. Considera-se satisfatório o tempo de seis segundos para extração e catorze para a publicação de um documento de 4.7MB, contendo 16 páginas.

Outra avaliação executada foi relacionada ao tempo de verificação. Esse tempo, consiste na busca (passo 2b da Figura 1), *download* (passo 4b da Figura 1) e a aplicação de um algoritmo mais eficiente (passo 5b da Figura 1), que

foi a utilização de uma adaptação do SIM. Os resultados estão apresentados na Figura 2(b).

O tempo de busca permanece basicamente o mesmo em todos os casos. O tempo de *download* varia de acordo com o tamanho do documento. Contudo, dois minutos para um documento de 4.7 MB pode ser considerado um tempo aceitável. A execução do algoritmo eficiente foi baixa, dada a característica simples do algoritmo. Porém, acredita-se que ao utilizar outros tipos de algoritmos mais trabalhosos, o tempo aumente.

Finalmente, foi testado o percentual de respostas corretas. Esta última avaliação demonstra a eficiência da detecção de plágio. Para tal, foram copiadas partes exatas de um documento em outro. Este processo foi realizado com cinco pares de documentos. O PeerDetect detectou todas as cópias, entretanto, quanto menor a parte copiada para outro documento, menor o grau de certeza apontado pelo PeerDetect. Os resultados estão apresentados na Tabela 2. Os graus de similaridade apresentados podem ser considerados baixos, porém vale ressaltar que o menor indício de similaridade deve ser conferido.

**Tabela 2.** Testes de grau de similaridade

| documento1 (palavras) | documento2 (palavras) | Tamanho da frase | Similaridade (%) |
|---|---|---|---|
| 3661 | 2217 | 10 | 0 |
| 3661 | 2217 | 50 | 4 |
| 12207 | 6094 | 100 | 2 |
| 8048 | 6094 | 400 | 10 |
| 12207 | 16094 | 2000 | 20 |

*Teste em um cenário real.* O primeiro teste real do PeerDetect foi realizado com uma professora na mesma universidade em que o sistema foi criado. Um dos autores tomou conhecimento de que ela havia descoberto que uma de suas alunas havia copiado um texto da Internet sem qualquer modificação. A professora evidenciou por conta própria duas fontes diferentes. Os autores então, solicitaram a ela o empréstimo desse trabalho para que fosse testado com o PeerDetect. Os resultados foram muito positivos e o sistema não só descobriu as duas fontes encontradas pela professora, como também detectou outras duas que a aluna havia utilizado para copiar outra parte do texto. A professora ficou muito satisfeita com o sistema, e espera que ele seja implantando como ferramenta oficial da universidade, como parte do sistema Eureka [14].

*Considerações Finais.* Em [15] é mencionada uma preocupação com a propriedade intelectual dos documentos acadêmicos. A preocupação diz respeito ao fato do autor do documento (o aluno, por exemplo) não autorizar legalmente a distribuição do documento para outras pessoas que não o seu professor. Ao fazer isto, dispondo do documento e enviando-o para outros servidores para que seja comparado com outros documentos como a maioria dos sistemas convencionais faz, estaria-se violando a propriedade intelectual do aluno. Dada esta preocupação, é possível adicionar a possibilidade do usuário escolher não passar a segunda fase. Desta forma o documento não seria copiado para uma análise mais eficiente, e as informações finais se baseariam somente nos resultados do primeiro algoritmo.

## 6 Conclusões

O principal objetivo deste trabalho é oferecer um sistema de detecção de plágio eficiente. A abordagem apresentada consiste em usar documentos disponibilizados em uma rede P2P como fonte de dados para detectar documentos fraudulentos. Este método utiliza um conjunto de tecnologias e mecanismos, como DHT e JXTA. A abordagem apresentada possui a vantagem de procurar em documentos indisponíveis na Internet.

Os autores estão dispostos a usar o PeerDetect como ferramenta oficial de detecção de plágio na Universidade, onde a cultura anti-plágio não é difundida e muitos professores desconhecem formas de como detectar plágio. A Universidade PUCPR [16] utiliza um sistema colaborativo de ensino denominado Eureka [14], através do qual professores e alunos possuem aulas virtuais. O sistema provê diversas funcionalidades, como entrega de trabalhos *online*, o que permite aos alunos enviar seus trabalhos via navegador e aos professores que os recebam também via navegador. Isto considera-se uma boa fonte de dados para o PeerDetect. Desta forma, caso o PeerDetect seja utilizado em sistemas similares em outras universidades, cada uma delas se tornaria um nó na rede P2P disponibilizando seus documentos.

## Referências

1. Zernike, K.: With student cheating on the rise, more colleges are turning to honor codes. New York Times **Novembro** (2002) 10–11
2. `www.google.com` (2008) Google Search Engine.
3. `www.zemoleza.com.br` (2008)
4. `www.10emtudo.com.br` (2008)
5. `www.canexus.com/eve/index.shtml` (2008)
6. `www.turnitin.com` (2008)
7. Traversat, B., Abdelaziz, M., Duigou, M., Hugly, J., Pouyoul, E., Yeager, B.: Project jxta virtual network (2002)
8. Balakrishnan, H., Kaashoek, M.F., Karger, D., Morris, R., Stoica, I.: Looking up data in p2p systems. Communications of ACM **46**(2) (2003) 43–48
9. Chen, X., Francia, B., Li, M., McKinnon, B., Seker, A.: Shared information and program plagiarism detection. IEEE Transactions on Information Theory **50**(7) (2004) 1545–1551
10. Finlay, S.: Copycatch. Master's thesis, University of Birmingham (1999)
11. Cheang, B., Kurnia, A., Lim, A., Oon, W.C.: On automated grading of programming assignments in an academic institution. Comput. Educ. **41**(2) (2003) 121–131
12. Wise, M.J.: Yap3: improved detection of similarities in computer program and other texts. SIGCSE Bull. **28**(1) (1996) 130–134
13. Ribeiro, H.B., Lung, L.C., Santin, A.O., Brisola, N.L.: Web2peer: A peer-to-peer infrastructure for publishing/locating/replicating web pages on internet. In: ISADS '07: Proceedings of the Eighth International Symposium on Autonomous Decentralized Systems, Washington, DC, USA, IEEE Computer Society (2007) 421–428
14. `eureka.pucpr.br` (2008)
15. Tosun, E., Wellington, B.: Detectit: a peer-to-peer plagiarism detection system. Corant Institute, NYU (2003) (Unpublished manuscript).
16. `www.pucpr.br` (2008) Pontifícia Universidade Católica do Paraná.