

Sistema que Recomenda Sistema: uma Solução para Recomendação de Componentes de Software

Tássia Camões Araújo ¹

¹Instituto de Matemática e Estatística
Universidade de São Paulo (USP)

EXAME DE QUALIFICAÇÃO DE MESTRADO

Programa: Ciência da Computação
Orientador: Prof. Dr. Arnaldo Mandel

Resumo. *A crescente expansão na oferta de serviços na rede mundial de computadores vem expondo cada vez mais seus usuários a inúmeras possibilidades de escolha. Esta visível pluralidade, somada à infinidade de interesses dos indivíduos envolvidos, acaba por demandar uma organização das informações de forma que seja possível aproximar destes usuários aquilo que supõe-se ser o que verdadeiramente necessitam. Tal suposição é auxiliada por sistemas computacionais de recomendação, que em geral utilizam o próprio comportamento do usuário como componente fundamental para decidir o que lhe deve ser apresentado como opções mais suscetíveis a aceitação. O presente trabalho propõe o desenvolvimento de um sistema que auxilia a recomendação de componentes de software. Vê-se em especial no universo dos softwares de código aberto uma enorme diversidade de opções que, pelo seu caráter majoritariamente não comercial, são apresentadas de forma menos ostensiva aos seus potenciais usuários. A distribuição Debian GNU/Linux reúne milhares de componentes de software nestas circunstâncias sob uma sólida infraestrutura, oferecendo assim a este trabalho uma série de benefícios técnicos para os experimentos. Esta distribuição será portanto utilizada, onde propõe-se a implementação e posterior análise comparativa das técnicas abordadas de recomendação, que por fim será objeto de uma consulta pública aos usuários acerca da sua eficácia, a ser também compilada e comentada na finalização do trabalho.*

1. Introdução

A popularização de recursos computacionais e do acesso à Internet nas últimas décadas proporcionou um aumento expressivo na quantidade de serviços e conteúdo à disposição dos usuários. Um dos fatores para este aumento é que os usuários, que anteriormente eram considerados meros consumidores, apresentam-se atualmente como produtores de conteúdo. [Castells 2006] analisa este fenômeno e afirma que a maioria da população acredita que pode influenciar outras pessoas, atuando no mundo através da sua força de vontade e utilizando seus próprios meios. Isto pode ser observado no surgimento e proliferação de serviços criados e mantidos pelos usuários: blogs, enciclopédias colaborativas, como a Wikipedia¹, repositórios para compartilhamento de fotografia e vídeo, como Flickr² e Youtube³, entre outros. Considerando a produção em termos de software, observa-se o exemplo das comunidades de software livre, que propiciam a construção coletiva de uma ampla gama de softwares de qualidade, em constante atualização e evolução, e organizado na forma de um *rossio* [Simon and Vieira 2008].

A grande diversidade de opções disponíveis nestes ambientes, apesar de positiva, representa uma sobrecarga de informações que pode confundir o usuário final. O indivíduo muitas vezes possui pouca ou nenhuma experiência pessoal para realizar escolhas em determinado contexto [Cazella et al. 2010]. Sendo assim, recomendações de outras pessoas são de grande utilidade, pois reduzem as dúvidas e auxiliam o processo de escolha dentre as muitas alternativas apresentadas. No entanto, diante do número de usuários e do volume de conteúdo que comumente deve ser considerado, recomendações no estilo “boca a boca” tornam-se ineficientes, pois exigem a comunicação direta entre os pares. Deste modo, a tecnologia passa a ter papel fundamental neste processo [Shardanand and Maes 1995].

Estratégias para a automatização de recomendações começaram a ser apresentadas pela comunidade acadêmica em meados da década de 90. O tema ganhou destaque com o crescimento do comércio eletrônico, onde apresentar o que o usuário tem interesse pode significar conquistar o cliente. Os Sistemas de Recomendação fazem a associação entre objetos de interesse e pessoas neles interessadas, filtrando as informações de forma a apresentar somente o que é relevante para o usuário [Adomavicius and Tuzhilin 2005]. Além da agilidade para encontrar o que se deseja, tais sistemas possibilitam a personalização de serviços e conteúdos, que são apresentados de maneiras distintas para usuários diferentes, a partir da identificação de interesses pessoais.

Este trabalho se insere no contexto de desenvolvimento de componentes de software, no qual os usuários são modelados como clientes e os componentes desenvolvidos como itens pelos quais os usuários têm interesse ou não. Assume-se que cada usuário tem um sistema operacional instalado e deve escolher quais aplicativos extras deseja obter para suprir suas necessidades pessoais. Diante da enorme quantidade de software disponíveis, nas mais diversas áreas de aplicação, configura-se um cenário onde um sistema de recomendação traria benefícios imediatos ao usuário, por auxiliá-lo a tomar decisões acerca da configuração do seu ambiente de trabalho.

Após uma breve introdução sobre sistemas de recomendação e principais es-

¹<http://wikipedia.org>

²<http://flickr.com>

³<http://youtube.com>

estratégias utilizadas, ao longo do texto são apresentadas soluções existentes para a recomendação de pacotes em distribuições GNU/Linux, como Debian, Fedora, Mandriva e Ubuntu. Em seguida é apresentada a proposta de implementação deste trabalho, que pretende aproximar os serviços atualmente disponíveis à comunidade ao estado da arte em sistemas de recomendação.

2. Sistemas de Recomendação

Os Sistemas de Recomendação aumentam a capacidade e eficácia do processo de indicação bastante popular nas relações sociais [Resnick and Varian 1997]. Tradicionalmente, as recomendações eram produzidas exclusivamente por especialistas na área em que se pretendia recomendar. Um exemplo muito comum é a recomendação de filmes em cartaz, publicada por críticos de arte nos principais jornais e revistas do país. Nos últimos anos, porém, percebeu-se que a opinião e o comportamento de usuários não especializados agregam valor às recomendações, que passaram a considerá-los amplamente. Seja explicitamente, quando o próprio usuário escreve sua opinião ou comentário, ou implicitamente, quando técnicas de recuperação da informação são utilizadas para analisar preferências e comportamento dos usuários, e partir destes gerar recomendações.

Segundo [Adomavicius and Tuzhilin 2005], o problema da recomendação pode ser formalizado da seguinte maneira: Seja C o conjunto de todos os usuários e S o conjunto de todos os itens que podem ser recomendados, como livros, filmes ou restaurantes. O espaço S de itens possíveis pode ser muito grande, variando de centenas de milhares até milhões de itens em algumas aplicações. Similarmente, o espaço de usuários também pode ser muito grande, milhões em alguns casos. Seja u uma função que mede a utilidade do item s para o usuário c , isto é, $u : C \times S \rightarrow R$, onde R é um conjunto ordenado (por exemplo, inteiros não negativos ou números reais num determinado intervalo). Então, para cada usuário $c \in C$, deve-se escolher o item $s' \in S$ que maximiza a função de utilidade. Mais formalmente, temos:

$$\forall c \in C, s'_c = \arg_{s \in S} \max u(c, s) \quad (1)$$

2.1. Desafios

Segundo [Vozalis and Margaritis 2003], os principais desafios de sistemas de recomendação são:

- *Qualidade das recomendações*: usuários querem recomendações nas quais eles possam confiar. Esta confiabilidade é alcançada na medida em que se diminui a incidência de falsos positivos, ou seja, recomendações que não interessam ao usuário;
- *Esparsidade*: a existência de poucas relações usuários-item por usuário resulta numa matriz de relacionamentos esparsa. Isto dificulta a localização de vizinhança, resultando em recomendações fracas.
- *Escalabilidade*: a complexidade do cálculo de recomendações cresce tanto no número de clientes quanto na quantidade de itens, portanto os algoritmos utilizados devem ser escaláveis.
- *Perda de transitividade de vizinhança*: usuários com comportamento semelhante a um determinado usuário u não necessariamente têm comportamento semelhante

entre si. Quando desejado, é necessário que se aplique métodos específicos para captura deste tipo de relação.

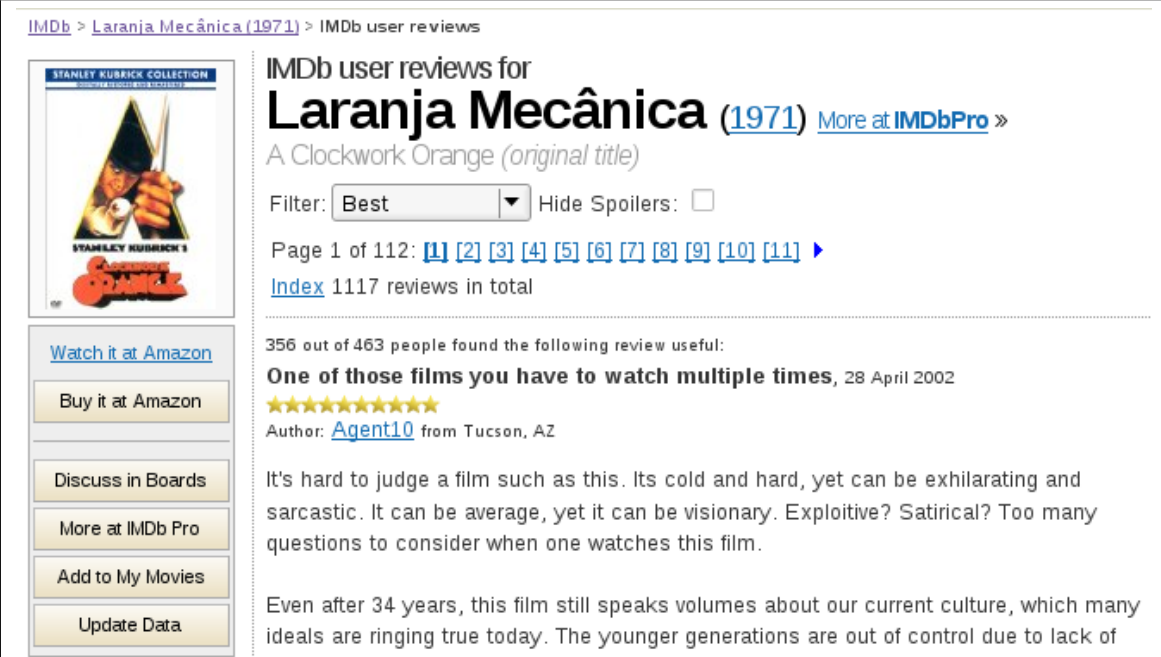
- *Sinônimos*: Quando o universo de itens possibilita a existência de sinônimos, a solução deve levar esta informação em conta para prover melhores resultados.
- *Problema da primeira avaliação*: Um item só pode ser recomendado se ele tiver sido escolhido por um usuário anteriormente. Portanto, novos itens precisam ter um tratamento especial até que sua presença seja "notada".
- *Problema do usuário incomum*: Indivíduos com opiniões que fogem do usual, que não concordam nem discordam consistentemente com nenhum grupo, normalmente não se beneficiam de seus sistemas de recomendações.

[Perfis de usuário e privacidade]

2.2. Estratégias de Recomendação

2.2.1. Reputação do produto

Bastante popular entre serviços de venda, como livrarias, sites de leilão e lojas de modo geral, esta estratégia consiste no armazenamento de avaliações dos produtos escritas por usuários e apresentação das mesmas no momento e local apropriado [Cazella et al. 2010]. A implementação desta solução é simples, visto que exige apenas a manutenção dos dados originais, não sendo necessária análise posterior alguma. No entanto, tem-se como premissa a imparcialidade dos usuários em suas opiniões, o que não pode ser verificado devido a seu caráter subjetivo e estritamente pessoal. Atualmente existem serviços especializados em reputação de produtos, que não têm venda associada, apenas disponibilizam as avaliações. É o caso do *Internet Movie Database* ⁴, apresentado na figura 1.



IMDb > [Laranja Mecânica \(1971\)](#) > IMDb user reviews

IMDb user reviews for
Laranja Mecânica (1971) [More at IMDbPro](#) »
A Clockwork Orange (original title)

Filter: Hide Spoilers: ☐

Page 1 of 112: [\[1\]](#) [\[2\]](#) [\[3\]](#) [\[4\]](#) [\[5\]](#) [\[6\]](#) [\[7\]](#) [\[8\]](#) [\[9\]](#) [\[10\]](#) [\[11\]](#) ▶
[Index](#) 1117 reviews in total

356 out of 463 people found the following review useful:

One of those films you have to watch multiple times, 28 April 2002
★★★★★★★★★
Author: [Agent10](#) from Tucson, AZ

It's hard to judge a film such as this. Its cold and hard, yet can be exhilarating and sarcastic. It can be average, yet it can be visionary. Exploitive? Satirical? Too many questions to consider when one watches this film.

Even after 34 years, this film still speaks volumes about our current culture, which many ideals are ringing true today. The younger generations are out of control due to lack of

[Watch it at Amazon](#)
[Buy it at Amazon](#)
[Discuss in Boards](#)
[More at IMDb Pro](#)
[Add to My Movies](#)
[Update Data](#)

Figura 1. Avaliação de usuário no IMDb

⁴<http://www.imdb.com/>

2.2.2. Associação

A partir da análise de uma base de dados que relaciona clientes e itens selecionados pelos mesmos, utiliza-se diversas técnicas para inferir associações entre os itens, como por exemplo: "Clientes que selecionaram os itens A, B e C também selecionaram o item D". As associações são caracterizadas por um suporte mínimo, que diz respeito à confiabilidade das mesmas. Portanto, quando uma associação é extraída, não significa que ela pode ser verificada em todos os casos, e sim numa porcentagem mínima, definida pelo suporte. A implementação desta estratégia é mais complexa, pois exige técnicas mais sofisticadas para identificação de padrões no comportamento dos usuários, principalmente quando se trata de um grande volume de dados. Um exemplo de aplicação desta abordagem é encontrado no site da Amazon (figura 2).



Figura 2. Recomendação por associação na Amazon

2.2.3. Baseada em conteúdo

Este método parte do princípio de que os usuários tendem a se interessar por itens semelhantes aos que eles já se interessaram no passado [Herlocker 2000]. O ponto chave desta estratégia é a classificação dos itens, por exemplo, através da identificação de atributos (autores e temas de livros, por exemplo). A partir dos atributos dos itens, aplica-se técnicas de recuperação da informação para definir a semelhança entre os mesmos. No caso de uma livraria, por exemplo, sugerir ao cliente outros livros do mesmo autor de livros previamente selecionados é uma boa estratégia.

Pelo fato de se apoiar na classificação dos itens, os resultados da recomendação são prejudicados nos casos em que os atributos não podem ser identificados de forma automatizada. Outro problema indicado por [Adomavicius and Tuzhilin 2005] é o da superespecialização, onde apenas itens similares aos já escolhidos pelos usuários são recomendados.

2.2.4. Colaborativa

Esta abordagem não exige a compreensão ou reconhecimento do conteúdo dos itens, pois se baseia na troca de experiências entre pessoas que possuem interesses em comum.

Define-se uma função que representa a distância entre os usuários, e a vizinhança de um usuário é composta pelos que estiverem mais próximos dele (com menor distância). Uma recomendação para o usuário u é produzida a partir da análise dos itens da vizinhança de u . Os itens que aparecerem com maior frequência na vizinhança farão parte da recomendação.

O problema da superespecialização é superado, visto que a recomendação neste caso não se baseia no histórico do próprio usuário, podendo apresentar itens totalmente inesperados. Outra contribuição é a possibilidade de formação de comunidades de usuários pela identificação de seus gostos e interesses similares [Cazella et al. 2010].

2.2.5. Híbrida

Consiste basicamente na combinação das abordagens colaborativa e baseada em conteúdo, unindo o melhor das duas técnicas e eliminando as fraquezas de cada uma [Cazella et al. 2010].

2.3. Técnicas

Nesta seção são apresentadas as técnicas que dão suporte às estratégias apresentadas na seção anterior.

- Tf-idf
- Classificador Bayesiano
- Vizinhos mais próximos (knn)
- Clustering
- Modelos probabilísticos

3. Distribuições GNU/Linux

Em 1983 Richard Stallman criou o projeto GNU⁵ com o objetivo de desenvolver um sistema operacional livre como alternativa ao UNIX⁶ – solução comercial amplamente difundida na indústria – e que fosse compatível com os padrões POSIX⁷ – família de normas definidas pelo IEEE com foco na portabilidade entre sistemas operacionais. Nos anos 90 o projeto GNU já havia atraído muitos colaboradores, que num curto espaço de tempo haviam desenvolvido inúmeros aplicativos para compor o sistema operacional. Porém, o núcleo do sistema (*GNU Hurd*) teve desenvolvimento mais lento.

Em outubro de 1991 o estudante finlandês Linus Torvalds publicou a versão 0.02 do Freax, o núcleo de um sistema operacional (*kernel*, em inglês) desenvolvido por ele na universidade. Nem o próprio Linus imaginava que aquele projeto, desenvolvido sem grandes pretensões, teria a dimensão do que hoje conhecemos como Linux [Torvalds and Diamond 2001].

Com o anúncio de Torvalds, Stallman vislumbrou a possibilidade de agilizar o lançamento do sistema operacional livre, se os aplicativos GNU que já estavam prontos fossem combinados com o núcleo recém-lançado – de fato, a primeira versão estável do

⁵<http://www.gnu.org>

⁶<http://www.unix.org/>

⁷<http://standards.ieee.org/develop/wg/POSIX.html>

GNU Hurd foi lançada apenas em 2001. Em 1992 o Linux foi licenciado sob a GNU GPL (*General Public License*⁸) e as equipes dos dois projetos começaram a trabalhar na adaptação do kernel Linux para o ambiente GNU. Este esforço conjunto possibilitou o surgimento das distribuições GNU/Linux, que são variações do sistema GNU/Linux.

Distribuições como Debian, Fedora, Mandriva e Ubuntu, oferecem diferentes soluções de sistema operacional compostas pelo GNU/Linux e mais uma gama de softwares selecionados pela comunidade ou empresa responsável por seu desenvolvimento. As distribuições reduzem a complexidade de instalação e atualização do sistema para usuários finais [Cosmo et al. 2008]. Os mantenedores da distribuição atuam como intermediários entre os usuários e os autores dos softwares (chamados de *upstreams*), através do encapsulamento de componentes de software em abstrações chamadas *pacotes*.

O processo de desenvolvimento e manutenção de uma distribuição varia bastante de uma para outra e está diretamente ligado à constituição do projeto. Quando são criadas por empresas, costumam receber colaboração dos usuários, mas de forma limitada, visto que as decisões chave são tomadas dentro da organização. Este é o modelo de desenvolvimento conhecido como *catedral* [Raymond 1999]. Por outro lado, os projetos criados independentemente, formam ao longo do tempo uma comunidade de desenvolvedores interessados em colaborar, e estes são os únicos responsáveis pelo sucesso ou fracasso do projeto. Neste modelo, que recebe o nome de *bazar*, o código-fonte está disponível durante todo o processo de desenvolvimento, não apenas nos lançamentos, permitindo que a contribuição seja mais efetiva. Nesses casos observa-se com mais clareza o fenômeno do consumidor produtor descrito anteriormente. Segundo Raymond este modelo é mais favorável ao sucesso, pois um bom trabalho de desenvolvimento de software é motivado por uma necessidade pessoal do desenvolvedor (ou seja, o desenvolvedor é também usuário).

A seleção e configuração dos aplicativos básicos de uma distribuição ficam a cargo da equipe que a desenvolve, com diferentes níveis de interferência da comunidade, como vimos acima. Este é um ponto chave no desenvolvimento, pois é um dos fatores que mais influenciam a escolha dos usuários pela distribuição. Um exemplo do impacto que tal seleção têm na comunidade foi a recente polêmica gerada pelo anúncio de Mark Shuttleworth, criador do Ubuntu, de que o projeto abandonaria o Gnome como interface padrão do usuário [Paul 2010]. No caso do Debian, uma decisão como esta seria resultado de longo período de discussão na lista *debian-desktop*. Polêmicas no Debian não são raras, por exemplo, quando o assunto é o tema do desktop padrão (papel de parede, cores e estilo das janelas, etc) que será lançado na próxima versão estável.

Softwares adicionais que atendem a necessidades específicas devem ser instalados pelo usuário ou administrador do sistema, após a configuração do sistema operacional. Este processo é facilitado pela infraestrutura de instalação de softwares provida pela distribuição (baseada em pacotes), mas a seleção dos programas fica a cargo do usuário.

3.1. Contexto do Trabalho

A escolha da distribuição na qual o desenvolvimento do trabalho seria baseado foi pausada pelos seguintes critérios: (1) Existência de um esquema consistente de distribuição de software; (2) Existência de dados estatísticos referentes ao uso de pacotes por usuários

⁸Suporte legal para a distribuição livre de softwares

e possibilidade de acesso aos mesmos; (3) Possibilidade de integração dos resultados do trabalho com os serviços disponibilizados pela distribuição; (4) Popularidade da distribuição. O Debian GNU/Linux foi selecionado pelos fatores apresentados a seguir.

1. O gerenciamento de pacotes em sistemas Debian GNU/Linux é realizado através do *APT (Advanced Packaging Tool)*. Ações como a busca, obtenção, instalação, atualização e remoção de pacotes são disparadas pelo APT, que num nível mais baixo faz uso do *dpkg*. O APT também gerencia conflitos entre pacotes, checando se novos pacotes a serem instalados conflitam com algum outro do sistema, e dependências, instalando todos os programas definidos como pré-requisitos antes da instalação de um pacote.
2. O *Popularity Contest*⁹ é o concurso de popularidade entre pacotes. Os usuários que aceitam participar do concurso enviam periodicamente a sua lista de pacotes instalados no sistema, que são armazenados no servidor do popcon. Diariamente as listas recebidas são processadas e dados estatísticos acerca do uso dos pacotes são gerados e disponibilizados no website do projeto.
3. Segundo o seu *contrato social perante a comunidade de software livre*¹⁰, o desenvolvimento do projeto Debian é guiado pelas necessidades dos usuários e da comunidade de software livre. Portanto, as iniciativas de colaboradores individuais, seja ele desenvolvedor oficial ou não, serão igualmente consideradas e passarão a fazer parte da distribuição se seguirem aqueles princípios e forem considerados de utilidade para a comunidade.
4. O Debian é um projeto de destaque no ecossistema do software livre. Desde o lançamento da primeira versão de sua distribuição, em 1993, o projeto cresceu bastante em termos de componentes de software (atualmente provê mais de 25.000 pacotes), colaboradores e usuários. A *Distrowatch*, que tem 671 distribuições em sua base de dados, classifica o Debian GNU/Linux entre as 10 distribuições mais populares¹¹. O Debian aparece na quinta posição em suas estatísticas de páginas visitadas¹². Já o *Linux Counter*¹³ apresenta o Debian como a segunda distribuição mais popular entre as máquinas cadastradas que rodam o kernel Linux (16%), ficando atrás apenas do Ubuntu (24%). Nas pesquisas da *W³Techs* sobre tecnologias para serviços web¹⁴, o Debian aparece em segundo lugar, estando presente em 27% dos servidores. Na primeira posição está o CentOS com 31%.

De maneira geral, quando o projeto Debian é mencionado trata-se não somente do sistema operacional, mas de toda a infra-estrutura de desenvolvimento e coordenação que dá suporte ao trabalho de cerca de 1500 desenvolvedores oficiais, além de outros milhares de colaboradores ao redor do globo. O trabalho é realizado de forma colaborativa, afinado pelo objetivo comum de produzir e disponibilizar livremente um sistema operacional de qualidade para seus usuários [Jackson and Schwarz 1998]. A interação entre os desenvolvedores acontece majoritariamente através da Internet, por meio de canais IRC e listas de discussão públicas. Não existe uma entidade formal ou qualquer tipo de

⁹<http://popcon.debian.org>

¹⁰http://www.debian.org/social_contract.pt.html

¹¹<http://distrowatch.com/dwres.php?resource=major>

¹²<http://distrowatch.com/stats.php?section=popularity>

¹³<http://counter.li.org/reports/machines.php>

¹⁴http://w3techs.com/technologies/history_details/os-linux

organização que concentre, coordene ou defina as atividades do projeto. O que observa-se é um modelo de governança consolidado que emergiu naturalmente ao longo de sua história [O'Mahony and Ferraro 2007].

Diante do exposto, optou-se pelo Debian GNU/Linux como ambiente de desenvolvimento, todavia, os resultados poderão facilmente ser adaptados para outros contextos, desde que as informações necessárias para o cálculo de recomendações estejam disponíveis (dados estatísticos).

4. Recomendação nas Distribuições

Diante da complexa e crescente estrutura do projeto Debian, observa-se um esforço por parte dos desenvolvedores, principalmente da equipe responsável pelo controle de qualidade¹⁵, de reunir, organizar e disponibilizar as informações ou meta-dados concernentes a esta estrutura [Nussbaum and Zacchiroli 2010]. A tabela 1 relaciona algumas destas iniciativas que estão diretamente ligadas ao gerenciamento de pacotes. Vale ressaltar que a maioria destas soluções foram inicialmente desenvolvidas num contexto extra-oficial, mas a medida que se mostraram úteis e eficientes foram absorvidas pela comunidade de usuários e desenvolvedores.

Tabela 1. Soluções de recomendação no Debian

<i>Solução</i>	<i>Descrição</i>	<i>Estratégia de recomendação</i>
BTS	Sistema de rastreamento de bugs, que é alimentado pelos usuários	Reputação
Popcon	<i>Popularity Contest</i> publica diariamente o resultado do concurso de popularidade entre pacotes	
Packages.qa	Criado pelo time de qualidade, reúne informações relativas à manutenção do pacote	
UDD	<i>Ultimate Debian Database</i> é outra iniciativa do time de qualidade, que reúne informações de diversos aspectos do Debian numa base de dados única. Usuários avançados podem consultar esta base para tomar decisões acerca de que pacotes usar.	
Debtags	Classificação de pacotes em produção, porém, sem esquema de recomendação	Baseada em conteúdo
Debtags	No cadastro de tags para pacotes, novas tags são sugeridas a partir das tags já associadas ao pacote	Associação

[Investigar iniciativas externas ao Debian: central de aplicativos ubuntu]

5. Metodologia

A execução deste trabalho será guiada pelo método de resolução de problemas de George Polya [Polya 1945], descrito passo-a-passo nas seções seguintes.

¹⁵<http://qa.debian.org>

5.1. Compreensão do Problema

Ao trazer o problema da recomendação para o contexto de distribuições GNU/Linux, considera-se que os componentes de software ou pacotes são itens e os usuários da distribuição são clientes. Em termos de entrada e saída do sistema de recomendação: Dada a lista de pacotes de um usuário específico (destinatário da recomendação), a partir da análise de listas de pacotes de outros usuários, deve-se retornar uma lista de pacotes recomendados, que indiquem pacotes de potencial interesse para o usuário destinatário.

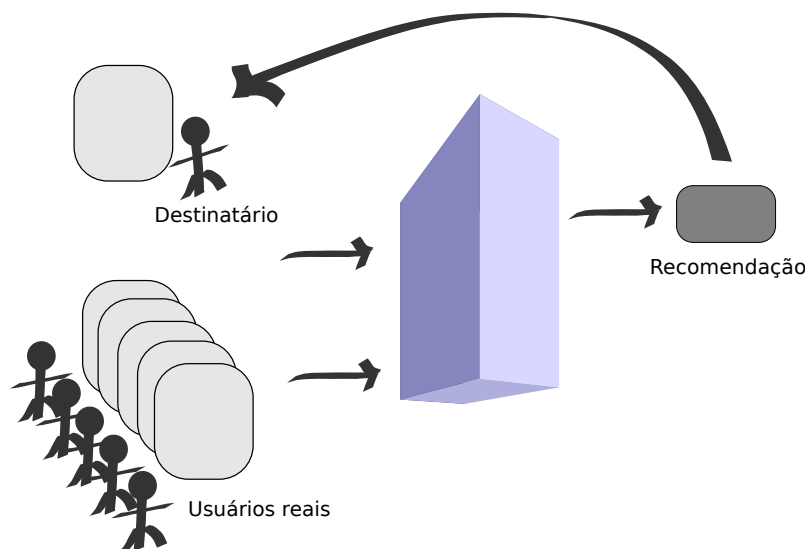


Figura 3. Cenário de uma Recomendação

Uma característica importante desta instância do problema de recomendação é que, diferentemente da situação usual onde os itens não se relacionam entre si (produtos na prateleira do supermercado, por exemplo), os componentes de software objetos desta pesquisa (pacotes debian) podem declarar explicitamente requisitos em seu conteúdo [Abate et al. 2009]. Requisitos positivos representam relações de dependência enquanto que os negativos representam conflitos, que podem ser definidos em diversos graus: dependência, sugestão, recomendação, conflito, substituição, quebra, etc. Por exemplo, se um componente a depende de b , significa dizer que b deve estar instalado no sistema para que a funcione como previsto. Por outro lado, se a conflita com c , a instalação de ambos os programas pode provocar um comportamento anômalo ou até comprometer o funcionamento de todo o sistema.

As relações entre os componentes são consideradas pelo sistema de gerenciamento de pacotes (no Debian, o APT), que recebe do usuário um pedido de modificar a instalação do sistema de alguma maneira – por exemplo, a instalação de um novo componente – e tenta satisfazer o pedido a partir do conhecimento de onde os componentes se encontram (repositórios de pacotes) e das relações entre os componentes. O gerenciador então promove a instalação de todas as dependências de um pacote antes de instalá-lo, e não permite a instalação de pacotes que conflitem com outros já instalados no sistema.

Portanto, o cálculo da recomendação de pacotes deve levar em conta as relações de dependência entre pacotes e desconsiderar pacotes dependentes nas buscas por associações, visto que as dependências de um pacote sempre (ou quase sempre) estão

presentes nos sistemas com tal pacote instalado.

Outro ponto a ser ressaltado é que as recomendações devem ser geradas a partir do comportamento do usuário (ação), e não da opinião. Neste trabalho não serão consideradas informações registradas por usuários no BTS ou em listas de discussão. Pretende-se calcular a recomendação a partir de dados do Popcon, que contém listas de pacotes instalados de milhares de sistemas em produção.

5.2. Estabelecimento de um Plano

O Popcon¹⁶ recebe periodicamente listas de pacotes instalados em sistemas de usuários reais para realização do concurso diário de popularidade entre pacotes. Esses dados porém devem ser pré-processados antes de serem considerados para o sistema de recomendação.

As listas de pacotes enviadas pelos usuários contém todos os pacotes do sistema, entre aplicativos, bibliotecas e até o kernel Linux. Para este estudo, apenas os aplicativos para usuário final são relevantes, pois são eles que devem compor a recomendação. Os pacotes que fazem parte do sistema básico também devem ser desconsiderados, pois em tese todos os usuários já os possuem visto que são essenciais para o funcionamento de qualquer sistema Debian. Assim pretende-se diminuir consideravelmente a ordem de grandeza das listas de pacotes, o que contribuirá para a eficiência do sistema.

Para o cálculo da recomendação pretende-se experimentar as estratégias:

- Baseada em conteúdo, a partir da classificação provida pelo debtags;
- Colaborativa, a partir dos dados do popcon;
- Híbrida, com uma mesclagem das duas abordagens.

Na implementação de tais estratégias, pretende-se utilizar algumas das técnicas mencionadas na seção 2.3 e fazer uma análise comparativa para avaliar qual delas é mais viável para o sistema em produção.

[Possibilidades de arquitetura para o serviço]

5.3. Execução do Plano

O presente trabalho será realizado entre os meses de janeiro e julho de 2001. Um planejamento de execução inicial é apresentado na tabela 2.

Tabela 2. Planejamento de execução

<i>Mês</i>	<i>Atividade</i>
Janeiro	Implementação das diferentes estratégias e técnicas
Fevereiro	
Março	Avaliação das soluções e lançamento do sistema para testes
Abril	Ajustes e melhorias na implementação
Maio	Preparação e realização de avaliação externa (feedback de usuários)
Junho	Escrita da dissertação
Julho	

¹⁶<http://popcon.debian.org>

5.4. Retrospectiva

Ao término da etapa de implementação pretende-se avaliar as soluções produzidas. Porém, devido à própria natureza do problema e dificuldade de medição de acurácia da saída, examinar os resultados obtidos por um sistema de recomendação não é tarefa trivial.

As métricas de avaliação mais utilizadas, herdadas da disciplina de recuperação da informação, são cobertura e precisão. Além dessas, serão consideradas também as métricas propostas por [Fouss and Saerens 2008]: robustez e novidade. Todas essas métricas, porém, partem do pressuposto de que para cada conjunto de dados avaliado existe uma recomendação correta, ideal, com a qual a saída do sistema deve ser comparada. Na inexistência deste ideal, [Vozalis and Margaritis 2003] sugere que sejam criados conjuntos de dados artificiais especialmente para a avaliação do sistema. Ainda assim não se garante que esse ideal artificial de fato representa a realidade.

Uma nova etapa de avaliação será realizada quando o sistema estiver em produção. Pretende-se aplicar questionários para usuários selecionados, com o intuito de obter um retorno mais fiel à realidade, ainda que de caráter mais subjetivo do que a primeira avaliação.

6. Conclusão

Referências

- [Abate et al. 2009] Abate, P., Boender, J., Cosmo, R. D., and Zacchiroli, S. (2009). Strong dependencies between software components. Technical report, MANCOOSI - Managing the Complexity of the Open Source Infrastructure.
- [Adomavicius and Tuzhilin 2005] Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749.
- [Castells 2006] Castells, M. (2006). A era da intercomunicação. *Le Monde Diplomatique Brasil*, Agosto 2006.
- [Cazella et al. 2010] Cazella, S. C., Reategui, E. B., and Nunes, M. A. (2010). A ciência da opinião: Estado da arte em sistemas de recomendação. In *JAI: Jornada de Atualização em Informática da SBC*, pages 161–216.
- [Cosmo et al. 2008] Cosmo, R. D., Zacchiroli, S., and Trezentos, P. (2008). Package upgrades in foss distributions: details and challenges. In *Proceedings of the 1st International Workshop on Hot Topics in Software Upgrades*, pages 7:1–7:5. ACM.
- [Fouss and Saerens 2008] Fouss, F. and Saerens, M. (2008). Evaluating performance of recommender systems: An experimental comparison. *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*.
- [Herlocker 2000] Herlocker, J. L. (2000). *Understanding and improving automated collaborative filtering systems*. PhD thesis.
- [Jackson and Schwarz 1998] Jackson, I. and Schwarz, C. (1998). *Debian Policy Manual*. Disponível em <http://www.debian.org/doc/debian-policy/>.
- [Nussbaum and Zacchiroli 2010] Nussbaum, L. and Zacchiroli, S. (2010). The ultimate debian database: Consolidating bazaar metadata for quality assurance and data mining. *IEEE Working Conference on Mining Software Repositories*.

- [O'Mahony and Ferraro 2007] O'Mahony, S. and Ferraro, F. (2007). The emergence of governance in an open source community. *Academy of Management Journal*, 50(5):1079–1106.
- [Paul 2010] Paul, R. (2010). Shuttleworth: Unity shell will be default desktop in ubuntu 11.04. *Disponível em* <http://arstechnica.com/open-source/news/2010/10/shuttleworth-unity-shell-will-be-default-desktop-in-ubuntu-1104.ars>.
- [Polya 1945] Polya, G. (1945). *How to Solve It: A New Aspect of Mathematical Method*. Princeton University Press.
- [Raymond 1999] Raymond, E. S. (1999). *The Cathedral & the Bazaar*. O'Reilly Media.
- [Resnick and Varian 1997] Resnick, P. and Varian, H. R. (1997). Recommender systems. *Communications of the ACM*, 40(3):56–58.
- [Shardanand and Maes 1995] Shardanand, U. and Maes, P. (1995). Social information filtering: Algorithms for automating word of mouth. In *Proceedings of the Conference on human factors in Computing Systems*.
- [Simon and Vieira 2008] Simon, I. and Vieira, M. S. (2008). O rossio não rival. *Disponível em* http://www.ime.usp.br/~is/papir/RNR_v9.pdf.
- [Torvalds and Diamond 2001] Torvalds, L. and Diamond, D. (2001). *Just for Fun: The Story of an Accidental Revolutionary*. HARPER USA.
- [Vozalis and Margaritis 2003] Vozalis, E. and Margaritis, K. G. (2003). Analysis of recommender systems algorithms. In *Proceedings of the 6th Hellenic European Conference on Computer Mathematics and its Applications*.