UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
INSTITUTO TÉRCIO PACITTI DE APLICAÇÕES E PESQUISAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

DIOGO MUNARO VIEIRA

AVALIAÇÃO DOS IMPACTOS DE SISTEMAS DE RECOMENDAÇÃO DE CONTEÚDO EM REDES P2P

Rio de Janeiro
2014

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
INSTITUTO TÉRCIO PACITTI DE APLICAÇÕES E PESQUISAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

DIOGO MUNARO VIEIRA

AVALIAÇÃO DOS IMPACTOS DE SISTEMAS DE RECOMENDAÇÃO DE CONTEÚDO EM REDES P2P

Dissertação de Mestrado submetida ao Corpo Docente do Departamento de Ciência da Computação do Instituto de Matemática, e Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para obtenção do título de Mestre em Informática.

Orientador: Carla Amor Divino Moreira DELGADO
Co-orientador: Daniel Sadoc MENASCHÉ

Rio de Janeiro
2014

CBIB VIEIRA, Diogo Munaro

Avaliação dos Impactos de Sistemas de Recomendação de Conteúdo em Redes P2P / Diogo Munaro VIEIRA. – 2014.
96 f.: il.

Dissertação (Mestrado em Informática) – Universidade Federal do Rio de Janeiro, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Programa de Pós-Graduação em Informática, Rio de Janeiro, 2014.

Orientador: Carla Amor Divino Moreira DELGADO.
Orientador: Daniel Sadoc MENASCHÉ.

1. Rede. 2. Cache. 3. Proxy. 4. Par-a-Par. 5. P2P. 6. ICN. 7. Recomendação. 8. Multimídia. 9. QoE. 10. AVoD. – Teses. I. DELGADO, Carla Amor Divino Moreira (Orient.). II. MENASCHÉ, Daniel Sadoc (Orient.). III. Universidade Federal do Rio de Janeiro, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Programa de Pós-Graduação em Informática. IV. Título

CDD

DIOGO MUNARO VIEIRA

Avaliação dos Impactos de Sistemas de Recomendação de Conteúdo em Redes P2P

Dissertação de Mestrado submetida ao Corpo Docente do Departamento de Ciência da Computação do Instituto de Matemática, e Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para obtenção do título de Mestre em Informática.

Aprovado em: Rio de Janeiro, _____ de _____ de _____.

_____

Profa. DSc. Carla Amor Divino Moreira DELGADO (Orientador)

_____

Prof. PhD. Daniel Sadoc MENASCHÉ (Orientador)

_____

Profa. DSc. Jonice de Oliveira Sampaio

_____

Prof. PhD. Paulo Henrique de Aguiar Rodrigues

_____

Profa. DSc. Giseli Rabello Lopes

_____

Prof. DSc. Antonio Augusto de Aragão Rocha

Rio de Janeiro
2014

Dedico esse trabalho a minha família e minha noiva que nunca me deixaram desistir de terminar o mestrado.

AGRADECIMENTOS

Agradeço aos meus pais por terem me orientado e educado durante toda minha vida, sempre me incentivando a tomar o melhor caminho.

Agradeço à minha noiva, ao meu irmão e ao meu cachorro pela paciência que tiveram quando não tive tempo para passar com eles.

Agradeço novamente à minha noiva pelo apoio que sempre me dá para continuar trilhando um bom caminho junto dela.

Agradeço aos meus orientadores Daniel e Carla pela paciência e dedicação ao enfrentar o desafio de orientar um biofísico na área de informática.

Agradeço novamente aos meus orientadores por toda ajuda e compreensão quando precisei trabalhar e realizar o mestrado concomitante.

Agradeço aos meus chefes e sócios pela compreensão que tiveram por estar cursando o mestrado junto com o trabalho.

Agradeço ao CNPQ e a Capes pelo fomento durante meu mestrado.

Agradeço à UFRJ e ao PPGI por me dar a oportunidade de crescer profissionalmente, tanto pelo mestrado quando pela ajuda de custo fomentada aos congressos da SBRC.

Agradeço aos organizadores da SBRC por terem gostado de meu trabalho e me escolhido para apresentar.

Agradeço a Deus e a todos os meus amigos encarnados e desencarnados que sempre querem meu bem.

RESUMO

VIEIRA, Diogo Munaro. **Avaliação dos Impactos de Sistemas de Recomendação de Conteúdo em Redes P2P**. 2014. 82 f. Dissertação (Mestrado em Informática) - PPGI, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2014.

Os sistemas de recomendação de conteúdo são uma forma transparente de conseguir extrair informações dos usuários e encaminhá-los aos seus possíveis desejos. Com o surgimento de novas tecnologias, aparelhos que acessam a internet e os sistemas AVoD (áudio e vídeo sob demanda), a filtragem de conteúdo para cada usuário surge como uma possível forma de modificar a QoE (qualidade de experiência) com que esses usuários obtém conteúdo de um serviço. Este trabalho propõe levar em consideração o ambiente de distribuição de conteúdo na recomendação e utiliza como foco do estudo redes P2P, que tradicionalmente, são utilizadas para transferência de arquivos. Ao consumir conteúdo multimídia nem sempre os usuários conseguem a QoE que desejam por não saberem o quão alto é o custo de disseminação do conteúdo a ser servido. Para mitigar essas questões, nesta dissertação é avaliado teoricamente como sistemas de recomendação de conteúdo podem afetar a QoE (custo de disseminação) com que um conteúdo pode ser servido em redes P2P, visando possibilitar maior flexibilidade para administradores de rede e usuários sobre o que desejam transmitir e receber, respectivamente. Para sugerir um modelo matemático que misture recomendação de conteúdo e QoE são propostas heurísticas que descrevem esse problema fazendo análises em cima da base de dados do MovieLens. O modelo criado é expansível a outras adaptações de heurísticas para diferentes cenários.

Palavras-chave: Rede, Cache, Proxy, Par-a-Par, P2P, ICN, Recomendação, Multimídia, QoE, AVoD.

ABSTRACT

Impact Assessment for Content Recommendation Systems on P2P Networks

VIEIRA, Diogo Munaro. **Avaliação dos Impactos de Sistemas de Recomendação de Conteúdo em Redes P2P**. 2014. 82 f. Dissertação (Mestrado em Informática) - PPGI, Instituto de Matemática, Instituto Tércio Pacitti, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2014.

Companies are always searching new ways to get their customers glad. Recommendation systems focus on identification of these customers preferences and improve their experience with a company service. New technologies with internet access, like cellphones are using AVoD (audio and video over demand) systems that use a lot of network bandwidth. Recommendation systems could change service's QoE (quality of experience) recommending a content with lower dissemination cost. Nowadays, P2P networks has a lot of users on file share applications, but this work focus on P2P network and how recommendation systems interact with them on multimedia applications. Recommendation systems need to know how much dissemination cost a file have and advise users about this cost when recommending this file. With a system like this, network administrators and common users could choice between files that want to transmit and receive. In this work we propose a new mathematical model and optimize mathematical heuristics that describes content recommendation and QoE of P2P networks using MovieLens dataset for analysis. This is a scalable model and accepts other heuristics for new scenarios.

Keywords: Network, Proxy, Cache, P2P, Peer-to-Peer, ICN, Recommendation, Multimedia, QoE, AVoD.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

QoE	*quality of experience*
SR	sistema de recomendação
AG	algoritmo genético
AVoD	áudio e vídeo sob demanda
ICN	*information centric network*
P2P	*peer-to-peer*
ROC	*rede orientada a conteúdo*

SUMÁRIO

1 INTRODUÇÃO

Sistemas de recomendação (SR) e de distribuição de conteúdo estão cada vez mais presentes no dia-a-dia de milhões de usuários na Internet. Embora tais sistemas tenham impacto direto no desempenho do sistema de distribuição de conteúdo e na própria disponibilidade do conteúdo, a relação entre os dois é ainda muito pouco compreendida. Com esses sistemas, é possível automatizar o trabalho de identificar as preferências dos usuários e colocá-los mais próximos de seus desejos (ADOMAVICIUS; TUZHILIN, 2005; BURKE, 2002). Essa automação consegue trazer mais vendas ao dono de uma loja virtual (SPIEGEL; KUNEGIS; LI, 2009; STORMER; WERRO; RISCH, 2006; SATYANARAYANA; RAJAGOPALAN, 2007), ou simplesmente mais utilizadores para um serviço de compartilhamento multimídia (DAVIDSON et al., 2010; MEI et al., 2007).

Como os SRs conseguem obter informações valiosas, são sempre implementados de forma transparente ou de fácil uso para o usuário (vide Figura 1.1), possibilitando que o próprio usuário contribua para melhores recomendações. Para recomendação de músicas, o que muitos sites como *Grooveshark* ou *LastFM* fazem é a combinação da análise do perfil do usuário com o que estão consumindo no momento, visando gerar recomendações com algoritmos cada vez mais eficientes. Esses algoritmos podem ser usados em multimídia e têm sido alvo de estudo na indústria por empresas como a Samsung (CHAKOO; GUPTA; HIREMATH, 2008), Netflix (NETFLIX, 2013), Google (DAVIDSON et al., 2010) e Microsoft (MEI et al., 2007). Embora a satisfação dos usuários dependa fortemente do conteúdo que lhes é recomendado, já que muitos usuários aderem às recomendações (ZHOU; KHEMMARAT; GAO, 2010), a qualidade de experiência (QoE) é outro fator chave que impacta a percepção dos usuários sobre o sistema (SITARAMAN, 2013).

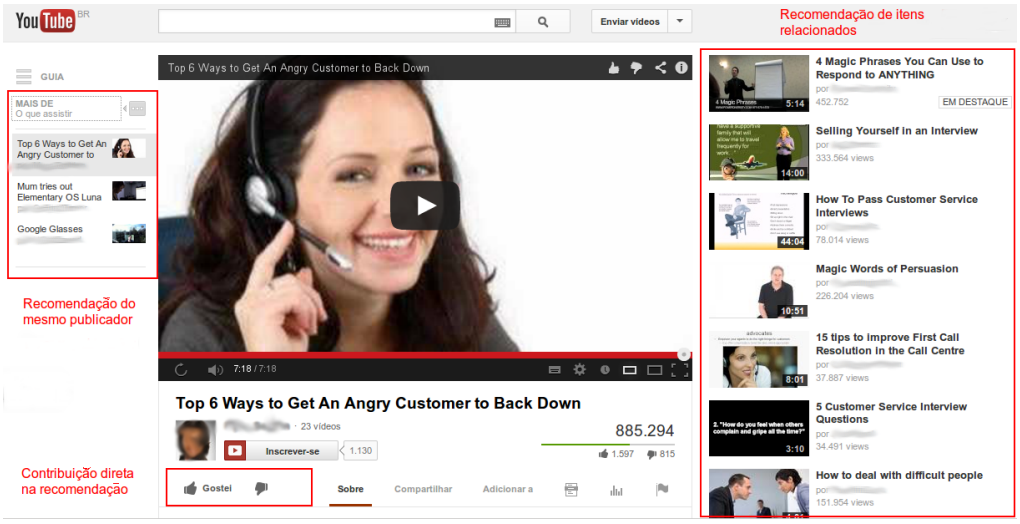


Figura 1.1: Funcionamento da recomendação de conteúdo no Youtube. O sistema de recomendação de conteúdo do Youtube possibilita que o usuário interaja contribuindo ao dizer se gostou ou não do vídeo, assim como possibilita verificar outros vídeos do mesmo publicador à esquerda e vídeos com conteúdo relacionado (recomendações do sistema para o usuário) à direita da imagem.

A satisfação dos usuários de um sistema de transmissão de conteúdo está intimamente relacionada com a QoE oferecida pelo sistema. Algumas métricas clássicas que influenciam na QoE são: atraso (tipicamente medido em milissegundos), *jitter* (variação do atraso dos pacotes) e custo de disseminação do conteúdo. De acordo com (SITARAMAN, 2013; KRISHNAN; SITARAMAN, 2012) o aumento de 2 segundos no atraso de início do vídeo resulta em 5,8% a mais de desistência dos usuários; um tempo de *rebuffer* (interrupção no vídeo para carregar mais) em 1% do tempo total do vídeo diminui a quantidade de tempo que o vídeo será assistido em 5,02%; e a taxa de abandono (probabilidade do usuário não voltar) em um site aumenta 2,32% se ele esteve indisponível, mesmo que durante um breve momento.

Uma área de pesquisa que se aproxima do problema estudado aqui é HCM (Multimídia Centrado nos Humanos) (CERQUEIRA et al., 2014; ELGAMMAL,

2006), mas essa área só prioriza aspectos para avaliação de QoE, sem inclusão das preferências do usuário. Alguns dos campos de estudo de HCM são: a forma de entrega do conteúdo (caso o usuário esteja usando celular ou computador), codecs utilizados, priorização de pedaços do conteúdo multimídia que são mais importantes e a otimização de outros parâmetros de QoE.

Mesmo com a comprovação de que os mecanismos de recomendação possuem grande influência na decisão de consumo - principalmente multimídia (em média 30% das visualizações de cada vídeo do YouTube é oriunda de recomendação (ZHOU; KHEMMARAT; GAO, 2010)) - nenhum algoritmo de recomendação foi até então usado para influenciar a QoE fornecida pela rede e nenhum artigo foi publicado em redes P2P (par-a-par). Somente poucas implementações de sistemas de *cache* usando as preferências dos usuários foram feitas durante a execução desse trabalho (vide Apêndice A).

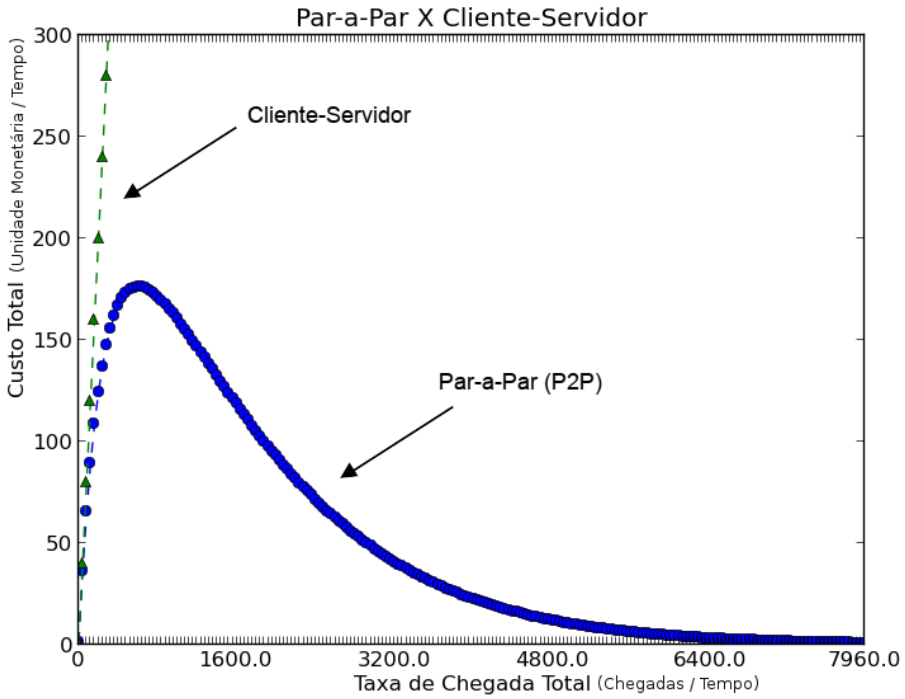


Figura 1.2: Comparativo entre custos ao aumentar a taxa de chegada para ambientes P2P e Cliente-Servidor. Lembrando que o custo P2P na comparação é o custo que um servidor tem para manter a rede. O custo em P2P estabiliza e diminui cada vez mais conforme aumenta a taxa de chegada.

Neste trabalho, foi estudada a inter-relação entre sistemas de recomendação de conteúdo e custo para disseminação desse conteúdo. Para esse estudo foi escolhido introduzir o ambiente P2P colaborando com a disseminação de conteúdo de servidores em detrimento de uma arquitetura de simples cliente-servidor, pois, ao introduzir a arquitetura P2P, o custo de disseminação de conteúdo dos servidores é menor por utilizar a taxa de *upload* dos *peers* da rede (Figura 1.2).

Sendo assim, o custo de disseminação considerado nesse trabalho abrange somente o custo que um servidor tem para manter uma rede P2P. Com isso, é con-

siderado que quanto maior o custo para disseminação de um determinado conteúdo, mais difícil será manter sua QoE.

Para integração do SR com essa rede P2P, foram utilizados sistemas de recomendação de conteúdo que possam ser afetados pelo custo de disseminação. Com isso apareceram as seguintes perguntas:

(i) Como o sistema de recomendação de conteúdo pode afetar o custo de disseminação em redes P2P?

(ii) Como o custo de disseminação de conteúdo em redes P2P deve ser levado em consideração pelo sistema de recomendação?

Tendo como objetivo a resposta dessas perguntas, foi estudada a relação entre custo e mecanismos de recomendação, construindo dois simuladores para três sistemas: P2P, ICN e *proxy cache*, mas somente o modelo P2P foi aprofundado. Esses simuladores foram implementados com algum parâmetro extra que possibilite acoplar interferências das preferências dos usuários na distribuição de conteúdo. Com isso, é possível obter alguns resultados para compreensão do problema e construção de um modelo analítico.

Ao construir esse modelo analítico fez-se necessário determinar a melhor forma do SR modificar as recomendações dadas aos usuários do sistema. Para encontrar a melhor modificação do SR em redes P2P foram utilizadas heurísticas e algoritmos genéticos (AG), tentando diminuir o custo de disseminação do conteúdo e aumentar a satisfação que o usuário tem ao receber uma recomendação.

Buscando aumentar a satisfação dos usuários e ao mesmo tempo melhorar o desempenho da rede, esse trabalho propõe algoritmos e soluções de recomendação

de conteúdo que levem em conta parâmetros de rede P2P. Nele é verificada a relação entre qualidade da disseminação de conteúdo em redes P2P e a recomendação desse conteúdo, assim propondo novos métodos para melhorar a qualidade de experiência (QoE) que os usuários recebem dessas redes P2P e a satisfação desses usuários. São implementadas também ferramentas para uma análise posterior em redes ICN e sistemas de *proxy cache*.

A Figura 1.3 ilustra o ambiente de estudo utilizado nesse trabalho para compreensão da implicação que SRs têm em redes P2P e, reciprocamente, o efeito que estas redes proporcionam nos SRs. Esse mesmo ambiente pode ser aproveitado para trabalhos futuros com redes ICN.

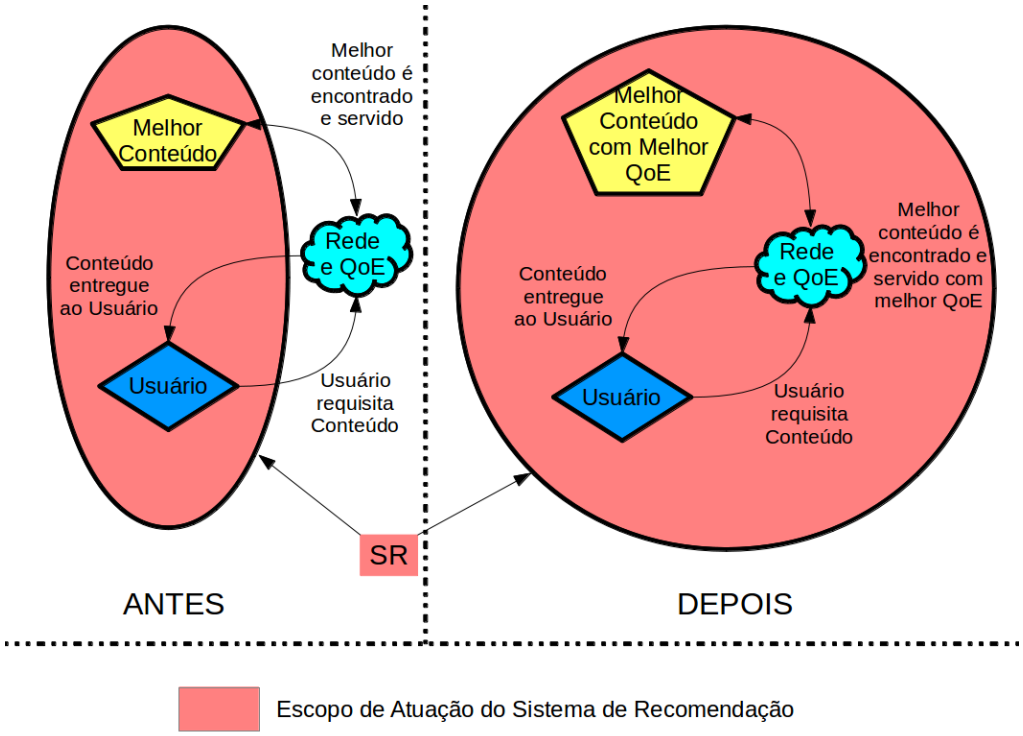


Figura 1.3: Simplificação do ambiente de estudo utilizado no trabalho. Mudança de atuação do sistema de recomendação (SR) que além de ter como base a satisfação do usuário leva em conta a QoE (qualidade de experiência) que ele pode conseguir.

Esse trabalho foi desenvolvido da seguinte forma: primeiro foi feita uma introdução ao referencial teórico, descrevendo alguns aspectos das redes de computadores e algoritmos de recomendação no capítulo 2. Em seguida, é discutido sobre a metodologia utilizada no capítulo 3 e os resultados obtidos através das heurísticas e do AG no capítulo 4. No capítulo 5, foi resumido o trabalho nas conclusões, apresentadas as limitações junto com os trabalhos futuros. No Apêndice B foi apresentado como aproveitar esse trabalho para uma outra vertente de estudos com redes ICN e servidores *proxy cache* através de um simulador, enquanto no Apêndice C estão os dados mais detalhados dos resultados do AG utilizado.

A dissertação deu origem à dois artigos, sendo o primeiro (VIEIRA; DELGADO; MENASCHE, 2013) um estudo preliminar sobre o comportamento de redes P2P e SR; e o segundo (VIEIRA; DELGADO; MENASCHE, 2014) o modelo que relaciona SR e custo de disseminação de conteúdo em redes P2P.

2 REFERENCIAL TEÓRICO

Agora será descrito o referencial teórico necessário para entendimento do trabalho e, nesse referencial, foram incluídos alguns trabalhos relacionados para melhor assimilação da teoria envolvida. Esse capítulo começa com uma introdução a alguns mecanismos bastante usados para recomendação de conteúdo na seção 2.1, seguindo para uma introdução sobre disseminação de conteúdo em redes de computadores na seção 2.2 e finalizando com a junção entre SRs e disseminação de conteúdo no contexto multimídia na seção 2.3.

2.1 Mecanismos de recomendação de conteúdo

Os mecanismos de recomendação tentam automatizar a filtragem de dados buscados por um usuário a partir dos dados desse mesmo usuário ou de outros similares que o sistema já tenha incorporado. Esses dados costumam ser: visitas à determinadas páginas, dados geográficos, sexo, preferências, entre outros. Devido a essa propriedade, mecanismos de recomendação têm sido muito utilizados por empresas para tentar conhecer as preferências de seus consumidores, assim podendo encontrar sempre a melhor sugestão para um determinado cliente. Automatizar essa detecção de sentimentos e preferências nunca se mostrou uma tarefa simples, pois os seres humanos possuem necessidades e desejos inesperados, que - mesmo demorando - podem mudar com o tempo (CHAKOO; GUPTA; HIREMATH, 2008; ADOMAVICIUS; TUZHILIN, 2005).

Vários mecanismos de recomendação foram desenvolvidos para compreender melhor os desejos dos seres humanos. Os mecanismos mais populares são o Filtro

Baseado em Conteúdo (vide seção 2.1.1) e o Filtro Colaborativo (vide seção 2.1.2), que normalmente são usados combinados com outras técnicas (ADOMAVICIUS; TUZHILIN, 2005; BURKE, 2002; PALAU et al., 2004).

2.1.1 Filtro baseado em conteúdo

O Filtro Baseado em Conteúdo é um mecanismo que se baseia no conteúdo dos itens para recomendar um item semelhante aos que o usuário previamente mostrou interesse (ADOMAVICIUS; TUZHILIN, 2005; BURKE, 2002). Esse conteúdo no qual ele se baseia são elementos explícitos como nome, descrição, *tags*, categorização ou *rating* do item a ser recomendado.

A vantagem de implantar o Filtro Baseado em Conteúdo é que o sistema não necessita de muitos usuários, já que se baseia somente no histórico do que o usuário já acessou. Em contrapartida, um sistema assim precisa de itens bem descritos, com informação suficiente para categorizá-los (STORMER; WERRO; RISCH, 2006). Outro problema encontrado nesse tipo de recomendação é a sugestão de itens sempre muito parecidos, não dando ao usuário sugestões fora da área de interesse atual.

O melhor uso do Filtro Baseado em Conteúdo descrito até agora (ADOMAVICIUS; TUZHILIN, 2005; STORMER; WERRO; RISCH, 2006; BURKE, 2002) é em sistemas com poucos usuários e muita informação sobre os itens. Quando um usuário é novo em um sistema, o Filtro Baseado em Conteúdo possui muita utilidade, pois não existe muita informação sobre aquele novo usuário e outras técnicas como o Filtro Colaborativo, que serão apresentados a seguir, são pouco eficientes quando há pouca informação sobre o usuário.

2.1.2 Filtro colaborativo

Diferente do Filtro Baseado em Conteúdo, o Filtro Colaborativo é um outro mecanismo que não precisa do conteúdo dos itens e se baseia em grupos colaborativos. O objetivo da criação desses grupos é de tentar classificar usuários com preferências similares e fazer com que 'recomendem' produtos uns para os outros (STORMER; WERRO; RISCH, 2006). Dentro dessa categoria de mecanismos de recomendação, existe ainda uma divisão em duas outras subcategorias: **Baseado no Modelo** e **Baseado em Memória** (ADOMAVICIUS; TUZHILIN, 2005; BURKE, 2002; GHAZANFAR; PRÜGEL-BENNETT, 2010).

A ideia da técnica de Filtro Colaborativo Baseado no Modelo é de fazer predições de agrupamento de usuários - criando modelos - para só depois realizar as recomendações. Para criar os agrupamentos (modelos), utilizam-se técnicas como redes neurais (BURKE, 2002), *Singular Value Decomposition* (SVD) (GHAZANFAR; PRÜGEL-BENNETT, 2010), e outras (ADOMAVICIUS; TUZHILIN, 2005).

O Filtro Colaborativo Baseado em Memória é um mecanismo mais heurístico que faz predições de recomendações a partir de avaliações anteriores dadas pelos usuários e outros dados do sistema, mas sem agrupamento prévio. Para isso, utiliza algoritmos de correlação como a correlação de cossenos (ADOMAVICIUS; TUZHILIN, 2005).

Normalmente o Filtro Colaborativo Baseado no Modelo e o Filtro Colaborativo Baseado em Memória se complementam na implementação de sistemas de recomendação, em que, para situações que agrupamentos são viáveis, utiliza-se a técnica baseada em modelo. Quando agrupamentos não são viáveis, utiliza-se a baseada em memória.

Independente da técnica utilizada em sistemas de recomendação, existem desvantagens de se utilizar o Filtro Colaborativo. Um dos principais problemas é o do novo item ou novo usuário, em que um item pode ter poucas avaliações ou um usuário pode ter poucos itens que gosta (ADOMAVICIUS; TUZHILIN, 2005). Outra questão que ocorre devido a diversidade humana é a 'ovelha negra', em que os usuários são iguais em alguns aspectos, mas não necessariamente querem sempre o mesmo item (BURKE, 2002).

2.1.3 Outros possíveis mecanismos de recomendação

Existem muitos outros mecanismos possíveis para recomendação, dentre eles estão a classificação de *Naïve Bayes* (GHAZANFAR; PRÜGEL-BENNETT, 2010) (que pode inclusive ser usado no filtro colaborativo baseado em modelo), Demográfico, Baseado em Utilidade e Baseado em Conhecimento. (BURKE, 2002). A classificação de *Naïve Bayes* classifica uma lista de palavras para cada usuário ou item, podendo facilitar a divisão de grupos pelo filtro colaborativo (GHAZANFAR; PRÜGEL-BENNETT, 2010); o mecanismo demográfico baseia suas recomendações, dividindo os usuários demograficamente; o mecanismo baseado em utilidade permite recomendações através de simples perguntas de preferência respondidas pelo usuário ou qualquer outra abordagem que gere uma função de utilidade, que pode ser usada para classificar usuários; o mecanismo baseado em conhecimento baseia suas recomendações através de todas as informações conhecidas no sistema, tentando extrair preferências dos usuários.

Esses outros mecanismos de recomendação não são tão abordados por serem pouco escaláveis, estando pouco presentes nos sistemas como possíveis alternativas, diferente do Filtro Colaborativo e do Baseado em Conteúdo. O mais comumente utilizado é combinar essas técnicas para gerar uma recomendação mais apurada,

filtrando os outros mecanismos (BURKE, 2002).

2.1.4 Mecanismos híbridos

Os mecanismos híbridos de recomendação são a combinação entre dois ou mais desses mecanismos. Existe uma série de possíveis combinações de mecanismos previamente descritos, mas a mais comum é a de comutação, em que o sistema utiliza dois ou mais mecanismos para cada situação. Essa combinação nos permite compensar a deficiência encontrada em algumas abordagens para que o sistema funcione com melhor eficiência em qualquer ocasião. O mais comum desses mecanismos híbridos é a mistura de Filtro Colaborativo e Filtro Baseado em Conteúdo (ADOMAVICIUS; TUZHILIN, 2005; SPIEGEL; KUNEGIS; LI, 2009; STORMER; WERRO; RISCH, 2006; PALAU et al., 2004).

Existem limitações que esse tipo de combinação (Filtro Colaborativo com Baseado em Conteúdo) não consegue evitar. O caso mais comum é quando há uma mudança brusca de hábitos do usuário. Nessas situações, o sistema costuma demorar muito a detectar essas mudanças e, para isso, são propostos outros possíveis mecanismos híbridos ainda pouco difundidos e estudados (BURKE, 2002).

Independente do mecanismo utilizado, normalmente esses mecanismos de recomendação são utilizados tomando como base peculiaridades de uma entidade (usuário, conteúdo, etc) e recomendando algo que seja mais interessante para essa entidade naquele momento. O recente trabalho tenta melhorar esse ambiente, se preocupando também em como essa entidade receberá o elemento de seu interesse.

2.2 Disseminação de conteúdo em redes de computadores

Nesse trabalho, é proposta a utilização de mecanismos de recomendação para otimização de redes P2P e deixa em aberto a possibilidade de um estudo com relação ao melhor armazenamento de conteúdo em *proxy cache* e em nós de redes ICN. Com isso, se torna necessário maiores explicações sobre esses tipos de redes e sobre *proxy cache*.

2.2.1 Redes P2P

Redes P2P (par-a-par) possuem arquitetura distribuída e são bastante vantajosas em termos de economia de recursos por dividir a carga de banda e armazenamento utilizando seus *peers*. Os *peers* são cada elemento que constitui essa rede e eles contribuem tanto para o processo de consumo, quanto para a distribuição de recursos (KUROSE; ROSS, 2010; MENASCHE; MASSOULIE; TOWSLEY, 2013; KELLERER, 1998).

As redes P2P podem ser categorizadas em dois grandes grupos: as redes puras e híbridas (SCHOLLMEIER, 2001).

Puras

As redes P2P puras são redes que, além de respeitar o princípio básico de compartilhamento de recursos, não percebem nenhuma perda ao se remover qualquer *peer* aleatório delas (SCHOLLMEIER, 2001).

Um exemplo bastante conhecido de rede P2P pura é o protocolo Gnutella, em

que os *peers* podem compartilhar conteúdos do seu computador e buscar conteúdos pela rede sem a necessidade de um servidor central (SCHOLLMEIER; HERMANN, 2003).

Híbridas

As redes P2P híbridas precisam de uma entidade central para que os *peers* consigam compartilhar os recursos (SCHOLLMEIER, 2001).

Basicamente, o exemplo mais popular dessas redes é do protocolo BitTorrent, que conta com um *tracker* organizando os *seeders* (quem dissemina conteúdo) e *leechers* (quem recebe conteúdo) de cada conteúdo. Ao entrar nessa rede, o usuário se torna um contribuidor dela, podendo compartilhar arquivos. Nela, todos os *peers* trabalham na alocação, distribuição e obtenção de um arquivo, assim podem estar fazendo *upload* e *download* dele ao mesmo tempo. Um bom exemplo disso são as distribuições Linux, como o Ubuntu que são distribuídas via *download* convencional ou via protocolo BitTorrent (*torrent*).

Esses *peers* se organizam através de agrupamentos divididos por conteúdo. Esses agrupamentos são chamados de *swarm* e um *peer* pode estar em mais de um deles caso esteja compartilhando vários conteúdos diferentes (CAMARILLO, 2009).

A tecnologia P2P, apesar do seu maior uso em compartilhamento de arquivos, é aplicada em vários outros ambientes, desde AVoD (áudio e vídeo sob demanda) (WU; LUI, 2012; ZHOU; FU; CHIU, 2012; DACUNTO et al., 2010; DACUNTO; VINKO; SIPS, 2011) e ambientes de colaboração - como o *PECOLE* (SADDIK et al., 2007), *PHAC* (CABANI et al., 2007) e outros (GUPTA; KAISER, 2005; GUARNIERI; SILVA; VIEIRA, 2013; GUARNIERI et al., 2014; ZHANG et al., 2012;

TRESTIAN et al., 2014; BERKET; ESSIARI; THOMPSON, 2005; EL-SADDIK; RAHMAN; HOSSAIN, 2005; DRAIDI et al., 2010) - até em um dos comunicadores mais utilizados do mundo: o Skype (KUROSE; ROSS, 2010).

2.2.2 *Proxy cache*

Servidores *proxy cache* são servidores que centralizam o tráfego de uma rede e guardam em seu *cache* alguns dos itens requisitados por usuários daquela rede. Esses servidores são mecanismos comumente implantados em grandes empresas por reduzirem o número de requisições realizadas para fora de sua rede privada. Isso resulta na economia de banda para a empresa e menor tempo de resposta para o usuário final (KUROSE; ROSS, 2010).

Com a explosão da internet, esses servidores *proxy cache* têm se popularizado, pois podem ser usados para guardar objetos HTML e transferir requisições HTTP, funcionando assim como um cliente e servidor ao mesmo tempo. Um exemplo bastante popular e *OpenSource* desse tipo de servidor pode ser criado utilizando o programa *Squid*.

Como os *caches* desses servidores possuem tamanho finito, são necessárias tecnologias para decidir quem permanecerá no *cache* quando um novo item chegar. Isso é feito através de controle de filas (AGUIAR RODRIGUES, 2013; MENASCHE; RODRIGUES, 2013; KLEINROCK, 1976; JAISWAL, 1968) e as tecnologias mais comuns para esses servidores são RR (*Random Replacement*), FIFO (*First In, First Out*), LRU (*Least Recently Used*) e LFU (*Least Frequently Used*). No RR, os elementos guardados no *cache* são removidos de forma aleatória. Segundo FIFO o primeiro elemento que entra no *cache* do servidor será o primeiro a sair. Para o LRU, são analisados sempre os últimos acessos, e o elemento sem ser acessado a

mais tempo é removido do *cache* quando é necessário guardar um novo elemento. O LFU utiliza um modelo de frequências de acesso. As frequências de acesso de todos os arquivos são rastreadas, e o item que possui uma frequência menor de acessos é removido quando necessário (RAO, 1978; CHAN et al., 1999; KELLY; JAMIN; MACKIE-MASON, 2001).

De todas essas políticas de escalonamento, a que tende a obedecer melhor as preferências dos usuários são a LRU e a LFU, mas já foi visto que é possível melhorá-las utilizando preferências dos usuários ou fatores específicos de uma determinada rede privada (CHAN et al., 1999; KELLY; JAMIN; MACKIE-MASON, 2001).

2.2.3 Redes orientadas a informação (ICN)

ICN é um novo paradigma de redes em que os usuários buscam o que desejam ao invés de buscarem onde está o conteúdo, como é feito atualmente na internet (AHLGREN et al., 2012). Nesse contexto, roteadores e *hubs* poderiam armazenar conteúdo e distribuir, servindo como *caches*.

Para trocar o método tradicional de busca por IP por busca por conteúdo, várias propostas estão sendo feitas em todo o mundo. No caso, as mais populares são PSIRP, 4WARD, PURSUIT1 e SAIL na Europa e NDN, CCN e DONA nos Estados Unidos (AHLGREN; DANNEWITZ, 2011), sendo que todas seguem o ideal de que qualquer nó pode publicar e aderir (*publish/subscribe*) ao conteúdo que desejar (GHODSI; SHENKER; KOPONEN, 2011).

Dentro da área de estudo de ICN existem vários problemas técnicos, como: segurança dos dados e políticas inter-domínio (GHODSI; SHENKER; KOPONEN, 2011); e tecnológicos, como: dificuldade de armazenar a tabela de roteamento e

não existir um modo rápido de armazenar e recuperar esses dados dos dispositivos intermediários (roteadores, *hubs*, *switchs*, etc) (PERINO; VARVELLO, 2011).

2.3 Relação entre recomendação e serviço de rede no contexto multimídia

A necessidade de estudos por uma recomendação multimídia mais eficiente se tornou mais interessante após o início de transmissão de arquivos desse tipo via *WEB*. Como a internet utiliza em geral um serviço de entrega de melhor esforço, não é possível garantir qualidade de experiência sem utilizar técnicas adicionais para esse fim. Em prol disso, uma série de parâmetros são adicionados para tentar extrair ao máximo as informações de vídeos, músicas ou voz e recomendar o melhor para o usuário (CHAKOO; GUPTA; HIREMATH, 2008; DAVIDSON et al., 2010; MEI et al., 2007; HIRSCH; PEARCE, 2000; OH et al., 2009). Outro fator motivador dessas pesquisas é a grande quantidade de músicas e vídeos atualmente disponíveis *online*, que torna a distribuição - com qualidade de experiência - e recomendação multimídia cada vez mais difícil.

2.3.1 Difusão multimídia

A recomendação de conteúdo é um ótimo meio para filtrar informação, logo, apresenta-se como uma ótima alternativa para a sugestão de multimídia num modo geral, buscando encontrar o melhor vídeo/áudio para o usuário (vide seção 2.1). Com base nisso e na escalabilidade apresentada por redes P2P (vide seção 2.2.1), muitos trabalhos estão aparecendo, utilizando essas redes para disseminação de conteúdo multimídia com sistemas AVoD (HECHT et al., 2012; WU; LUI, 2012; ZHOU; FU; CHIU, 2012), mas nenhum deles se preocupou em verificar melhorias de QoE

utilizando SRs (vide Apêndice A).

Focando em difusão de áudio, são citados especificamente três trabalhos, sendo o primeiro criado em 2003 e descontinuado desde início de 2009 chamado P2P-Radio; e os outros dois, mais recentes, sendo um chamado Radiommender (HECHT et al., 2012) e o outro BTStream (MENDONÇA; LEÃO, 2012). P2P-Radio é um projeto de código aberto propondo uma rádio P2P, enquanto o segundo e o terceiro, como estado da arte, são, respectivamente, uma rádio P2P descentralizada com recomendação e um reprodutor de conteúdo multimídia usando BitTorrent em *streaming*. É visível que desde 2003 até os dias atuais, as propostas de disseminação multimídia permanecem crescendo, mas sem focar na possibilidade de melhoria do desempenho da rede com o uso de recomendação.

2.3.2 *Streaming*

O *streaming* é uma técnica de exibição de um arquivo em partes, sem precisar que esteja totalmente carregado. No caso de *streaming* de áudio ou vídeo *WEB*, uma das alternativas comumente utilizada é o pré-carregamento, de parte ou do conteúdo inteiro, em servidores *proxy* ou CDNs (*content delivery network*) próximos aos usuários para facilitar a transmissão desse conteúdo com pouco tempo de carregamento (WANG et al., 2004; BRADSHAW et al., 2005), visando melhor QoE.

Naturalmente, existe preocupação por parte das empresas que seus usuários consigam consumir esse conteúdo multimídia que está sofrendo *streaming* com o melhor tempo de carregamento possível para que não desistam enquanto esperam o armazenamento em *buffer* (SITARAMAN, 2013). Devido a essa preocupação e pela sua eficácia, técnicas de *streaming* são aplicadas desde *sockets* HTTP até AVoD, seja utilizando P2P ou cliente-servidor. Independente da tecnologia aplicada,

todos as tecnologias de *streaming* disponibilizam a mesma base para disseminação do conteúdo multimídia.

A utilização de *streaming* foi um fator chave para melhor difusão de conteúdo multimídia via internet. Como esse tipo de conteúdo costuma necessitar de bastante atenção com relação à qualidade com que são disponibilizados, foi o escolhido para uma análise de qualidade da rede nesse trabalho.

O conteúdo descrito nesse capítulo ajuda a compreender os desafios desse trabalho em juntar custo de disseminação, que atualmente é uma variável totalmente intrínseca da rede, com recomendação de conteúdo, que atualmente é somente relativa ao usuário. Foram apresentados também diversos ambientes, como P2P, *proxy cache* e ICN que o trabalho pode impactar, a fim de simplificar a compreensão da extensibilidade desse trabalho para *streaming* ou alocação e disseminação de conteúdo em vários ambientes.

3 PROPOSTA DE UM MODELO RELACIONANDO RECOMENDAÇÃO DE CONTEÚDO E O DESEMPENHO DA REDE P2P

Está descrito nesse capítulo o modelo proposto para entender as questões relativas a interdependência entre recomendação de conteúdo e desempenho da rede P2P. O objetivo desse modelo é permitir a flexibilização de ambas as partes para melhorar a satisfação do utilizador de um sistema feito a partir desse modelo. Com isso, os objetivos desse modelo podem ser divididos em responder duas grandes questões:

1. A satisfação do usuário ao consumir um conteúdo recomendado está relacionada ao conteúdo em si e também à qualidade com que este lhe é servido. Entrando no contexto multimídia, a exigência da QoE de alta qualidade aumenta, por ter a necessidade de transmissão sob demanda sequencial do conteúdo. Tendo isso como base, **como o sistema de recomendação de conteúdo pode fazer uso das métricas de redes para aumentar a satisfação do usuário?**

2. Classicamente, assume-se que a demanda de tráfego por cada conteúdo na rede é uma variável exógena (fora do controle do sistema). Porém, se considerar a recomendação como parte do sistema, essa prerrogativa não é mais válida. Sendo assim, **como o administrador de rede pode fazer uso do sistema de recomendação de conteúdo para aumentar o desempenho de redes P2P?**

As métricas utilizadas para estudar essas questões foram:

- identificar cenário mais apropriado para explorar modelo proposto;
- conseguir um modelo que possibilite medir desempenho da rede P2P;
- conseguir um modelo que consiga medir a satisfação do usuário em uma rede P2P;
- integrar modelos para conseguir maximizar o desempenho da rede e a satisfação do usuário;
- encontrar variáveis do modelo que o administrador de redes ou o sistema de recomendação possam interferir para adequar o modelo às suas necessidades;
- determinar parâmetros do modelo que precisem ser estudados para melhor integração.

Ao visualizar essas métricas e a necessidade de integrar um modelo que combine recomendação de conteúdo e desempenho de redes P2P, fica evidente a utilização de QoE, pois a experiência do usuário deve influenciar diretamente no quão satisfeito esse usuário estará com o conteúdo recebido. Caso a QoE seja ruim, o usuário ficará insatisfeito com o sistema, mesmo que o conteúdo recebido seja bom.

Como se fez necessário a utilização de QoE, é nítido um campo abrangente de estudo com os conteúdos multimídia por ser um campo mais crítico, já que o usuário que esteja usando VoD pode receber o conteúdo com mais ou menos qualidade durante sua experiência.

Durante a avaliação do modelo foram utilizados filmes, mas a adaptação desse modelo é válida para qualquer conteúdo multimídia. Essa avaliação e os resultados finais só foram analisados em cunho teórico, não sendo testados com usuários reais, ficando assim como um trabalho futuro.

3.1 Parametrização do sistema de recomendação de conteúdo via *traces*

Para começar o trabalho, fez-se necessário modelar minimamente o funcionamento de um sistema de recomendação (SR). Em um sistema AVoD que tenha um SR o usuário assiste determinado filme k e em seguida pode ser influenciado pelo SR a sair desse filme para assistir um outro filme i, com isso, é possível medir o quanto esse usuário gostou desses filmes através da coleta de avaliações, por exemplo. Essas avaliações ajudam a compreender o quanto um usuário se sentiu recompensado de ter consumido aquele conteúdo e o quanto o SR foi eficiente de ter recomendado aquele conteúdo. Com isso, são criadas inferências para outros usuários que possam ter gostado do filme k e se irá ou não gostar do filme i como satisfação de se ir do filme k para o filme i ($r_{k,i}$). Agora será apresentada uma abordagem simples para parametrizar $r_{k,i}$.

Para tal, foi assumido que tem-se em mãos um *trace* indicando, para cada usuário, os filmes que este já assistiu e o número de estrelas que o usuário atribuiu a cada filme. Tal informação está disponível sobre os usuários do sistema MovieLens (MOVIELENS, 2013), que contém não apenas as avaliações dos usuários mas também os instantes em que elas ocorreram. O *trace* do MovieLens possui mais de 100,000 avaliações de filmes, sendo que neste trabalho é amostrado de forma uniforme e aleatória 1,000 delas. Dessa forma, obtêm-se $N = 551$ filmes. O número de avaliações para cada filme é ilustrado na Figura 3.1.

Como o trabalho fez uma avaliação teórica sobre a junção de sistemas de recomendação de conteúdo e redes P2P utilizando como base de análise o banco de dados do MovieLens, parâmetros como precisão e revocação nos SR não foram utilizados, mas adaptados. Isso se deve ao fato que o MovieLens não promove uma interação direta do usuário com o conteúdo durante a avaliação - ao contrário do

que acontece no Netflix, por exemplo - e isso não promove o mesmo sentimento ao avaliar o conteúdo. Apesar disso, o MovieLens se mostrou uma base interessante por ter as avaliações abertas.

Para adaptação dos parâmetros de precisão e revocação, considera-se um *trace* no qual U usuários deram notas (*ratings*) a N filmes. Cada *rating* é um número de estrelas, entre 1 e 5 conforme a escala Likert (MATELL; JACOBY, 1971). Seja $\mathsf{rating}_{u,k,i}$ o *rating* do usuário u ao filme i, para um usuário u que tenha feito o trajeto do conteúdo k para o conteúdo i, ou seja, que avaliou o conteúdo k e posteriormente o conteúdo i, e $\mathsf{rating}_{u,k,i} = 0$ caso contrário. Seja $\mathsf{rating}_{u,i}$ o *rating* do usuário u ao filme i, caso o usuário u tenha avaliado o filme i, e $\mathsf{rating}_{u,i} = 0$ caso contrário. Seja $\mathsf{Nrating}_i$ o número de usuários que avaliaram o conteúdo i e $\mathsf{Nrating}_{k,i}$ a quantidade de usuários que fizeram o trajeto do conteúdo k para o conteúdo i e avaliaram i. Normalizando o número de avaliações a um filme k pelo total de avaliações, obtêm-se a taxa de chegada (popularidade) do filme k, λ_k,

$$\lambda_k = \Lambda \left(\mathsf{Nrating}_k / \sum_{i=1}^{N} \mathsf{Nrating}_i \right), \quad k = 1, \ldots, N \tag{3.1}$$

sendo Λ a taxa agregada de chegada de usuários ao sistema. No capítulo 4, será investigado o efeito de Λ nas diferentes métricas de interesse.

A partir do *trace* fornecido, calculam-se os valores de $r_{k,i}$ tomando como inspiração o modelo do PageRank (PAGE et al., 1999), e assim, foi adaptada a medição de precisão e revocação para

$$r_{k,i} = \left[\left(\frac{\sum_{u=1}^{U} \mathsf{rating}_{u,k,i}}{\mathsf{Nrating}_{k,i}} \right) \Big/ \left(\sum_{i=1}^{N} \frac{\sum_{u=1}^{U} \mathsf{rating}_{u,k,i}}{\mathsf{Nrating}_{k,i}} \right) \right] \beta + \frac{(1-\beta) 1_{k \neq i}}{N-1}, \quad (k,i) \in \{1, \ldots, N\}^2 \tag{3.2}$$

onde $1_{ki \neq i} = 1$ caso $k \neq i$ e 0 caso contrário.

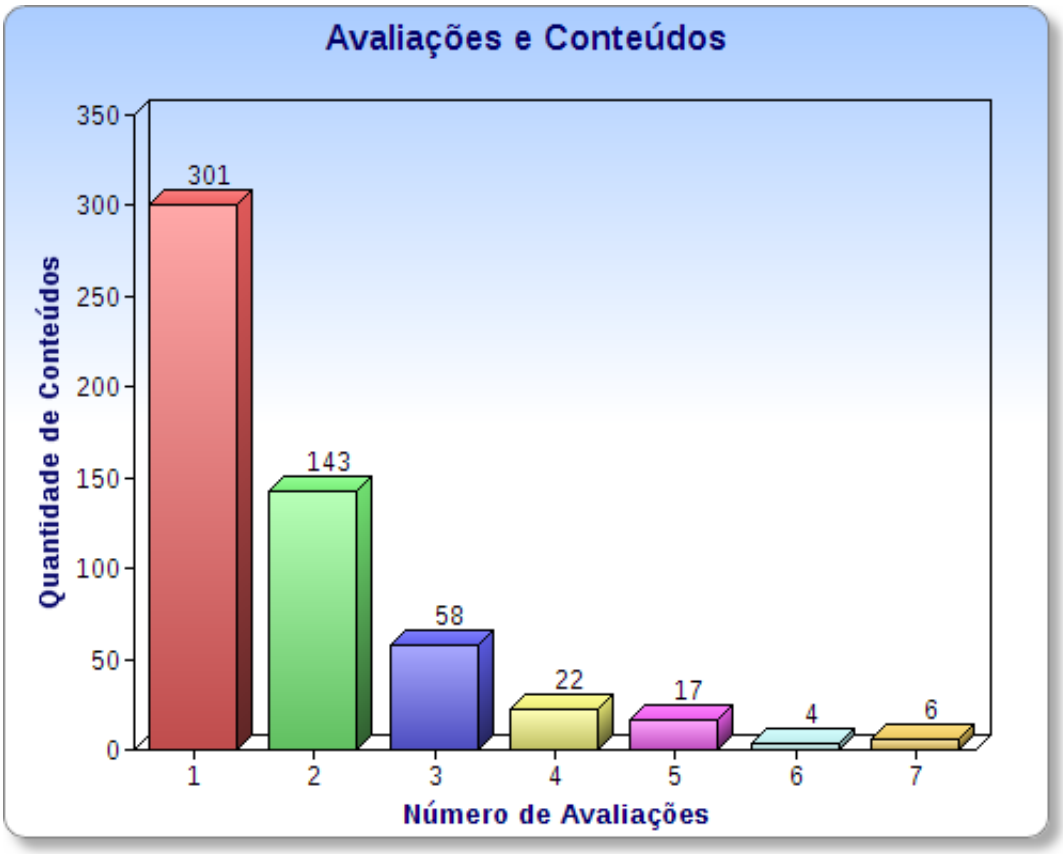


Figura 3.1: Número de avaliações por filme. O número de filmes para os quais $Nrating_i$ é igual a 1, 2, ..., 7 foi 301, 143, 58, 22, 17, 4 e 6 respectivamente.

O primeiro termo entre parêntesis na eq. (3.2) corresponde ao *rating* médio dado ao conteúdo i pelos usuários u que assistiram k e depois i, enquanto o segundo termo é um fator de normalização. O segundo termo é o somatório, para todos os arquivos i, $i = 1, \ldots, N$, do *rating* médio dado ao conteúdo i pelos usuários que assistiram k. Note que $0 \leq r_{k,i} \leq 1$.

Assim como no PageRank, foi assumido um fator de atenuação β. Nesse trabalho, foi considerado $\beta = 1$. Esse fator reflete a probabilidade do usuário continuar vendo e avaliando conteúdos, ao invés de simplesmente escolher assistir um filme aleatoriamente. Em trabalhos futuros deve ser considerado $\beta < 1$ para capturar o fato de que, mesmo que nenhum usuário tenha assistido um determinado par de filmes, a recomendação mútua possa ser positiva.

3.2 Modelo equacionando sistemas de recomendação e custo de disseminação

Nesta seção apresenta-se o modelo analítico para estudo dos *tradeoffs* envolvidos na recomendação de conteúdo levando em conta os custos e desempenho da rede P2P.

Primeiro, apresentam-se os parâmetros do modelo, as variáveis de controle e a metodologia para parametrização do modelo em função de *traces*. Em seguida, é introduzido como serão calculadas as funções de custo de disseminação de conteúdo e a recompensa que um usuário tem ao ter determinado conteúdo recomendado.

Parâmetro	**Descrição**
N	número de *swarms* (conteúdos)
$\mathsf{rating}_{u,i}$	avaliação do usuário u ao conteúdo i ($Pontos\ Likert$)
$\mathsf{rating}_{u,k,i}$	avaliação do usuário u ao conteúdo i ao vir do conteúdo k ($Pontos\ Likert$)
$\mathsf{Nrating}_{k}$	quantidade de avaliações dadas ao conteúdo k
$\mathsf{Nrating}_{k,i}$	quantidade de avaliações dadas ao conteúdo i por usuários vindos do conteúdo k
λ_k	popularidade do conteúdo k normalizada entre 0 e Λ ($\frac{Chegadas}{Tempo}$)
Λ	taxa agregada de chegada de usuários ao sistema ($\frac{Chegadas}{Tempo}$)
$r_{k,i}$	recompensa por recomendar-se i para o requisitante de k ($Pontos\ Likert$)
q	probabilidade de aceitação de recomendação
ϵ	probabilidade do SR mudar o conteúdo que seria recomendado
β	fator de atenuação que permite que usuários consumam conteúdos ainda não avaliados
Variável	**Descrição**
$p_{k,i}$	probabilidade de recomendar-se conteúdo i para o requisitante do conteúdo k
Métrica	**Descrição**
R	recompensa total ($\frac{Pontos\ Likert}{Tempo}$)
C	custo total ($\frac{Unidade\ Monetária}{Tempo}$)
P	custo total perturbado pelo sistema de recomendação ($\frac{Unidade\ Monetária}{Tempo}$)

Tabela 3.1: Tabela de notação: há 11 parâmetros no modelo proposto, 1 variável de controle referente ao sistema de recomendação e 3 métricas de interesse.

3.2.1 Parâmetros e variáveis de controle

3.2.1.1 Parâmetros do sistema

Os parâmetros do sistema são dados coletados a partir de *traces* de um sistema já existente ou fixados em um determinado valor para análise em trabalhos futuros. A seguir, são descritos os três principais parâmetros considerados para obtenção dos resultados:

- Considera-se N arquivos (conteúdos) disseminados por meio de uma rede;
- A taxa de chegada λ_k ($k = 1,2,\ldots,N$) é também referenciada como a popularidade do conteúdo k (vide Tabela 3.1), já que a quantidade de acessos é proporcional à popularidade do conteúdo;
- Seja $r_{k,i}$ a satisfação (recompensa) que quem assistiu o conteúdo k tem ao assistir o conteúdo i. O parâmetro $r_{k,i}$ depende da relação entre os conteúdos k e i, e pode ser parametrizado via *traces* expandindo a explicação da seção 3.1 para qualquer conteúdo;

Além desses parâmetros principais, outro parâmetro que não foi explorado nesse trabalho é a probabilidade q de um usuário aceitar uma recomendação. O parâmetro q é assumido constante igual a 100% ao longo de cada um dos cenários analisados. Usuários que não aceitam uma recomendação simplesmente vão embora do sistema.

Um último parâmetro considerado foi ϵ, que é a probabilidade de se redirecionar uma recomendação de um conteúdo para outro. Esse parâmetro será melhor

explicado na seção 3.3 junto com as heurísticas, sendo que ele foi definido como 0,01 nesse trabalho.

Trabalhos futuros consistem em associar q à qualidade da recomendação e a verificar melhores valores de ϵ nas heurísticas.

3.2.1.2 Variáveis de controle

Considera-se $p_{k,i}$ a probabilidade do sistema de recomendação de conteúdo recomendar o conteúdo i para quem requisitou e consumiu o conteúdo k, $k = 1, \ldots, N$, $i = 1, \ldots, N$. Deste ponto em diante, assume-se $p_{i,i} = 0$. O principal objetivo deste trabalho consiste na determinação de $p_{k,i}$ de tal forma a satisfazer o usuário e ao mesmo tempo levando em conta as características de rede P2P. Assume-se $p_{k,i}$ como variável de controle, pois é possível manipular essa probabilidade para mudar a recomendação padrão que o usuário receberia se ela não existisse.

3.2.2 Recompensas do sistema de recomendação de conteúdo

A recompensa em nosso modelo representa o quão satisfeito o usuário fica ao aceitar uma recomendação e essa medição pode ser obtida pelo próprio sistema de recomendação, conforme descrito em 3.2.1.

Dadas as recompensas individuais dos pares de filmes, a recompensa total do sistema de recomendação de conteúdo, R, é definida como

$$R = \sum_{k=1}^{N} \sum_{i=1}^{N} \lambda_k p_{k,i} q r_{k,i} \tag{3.3}$$

3.2.3 Custos do sistema de disseminação de conteúdo

Não foi possível identificar uma formulação genérica para o custo de disseminação nos três ambientes: P2P, ICN e *proxy cache*, mas é possível definir um modelo genérico que atenda as particularidades de cada ambiente e, ao mesmo tempo, abstrair essas particularidades em uma formulação geral. Assim, a seguir foi definida uma função de custo diferente para ambientes P2P e outra para ICN e *proxy cache*. Dessas definições, somente a função de custo de disseminação para ambientes P2P foi explorada durante o trabalho.

O único fator que pode ser aproveitado para outros sistemas que tenham taxa de chegada é o γ_i. Ele é a taxa de chegada perturbada para o conteúdo i, ou seja, a taxa levando em conta o impacto do sistema de recomendação. A taxa γ_i é dada, em função dos parâmetros do sistema, por

$$\gamma_i = \sum_{k=1}^{N} \lambda_k p_{k,i} q \tag{3.4}$$

Note que ao considerar a taxa de chegada ao conteúdo i (γ_i), leva-se em conta apenas as requisições provenientes de recomendação. Isso corresponde ao caso em que se conhece o histórico do usuário que, no passado, requisitou o conteúdo k, mas este não possui mais o arquivo k ao chegar ao sistema. Dessa forma, o usuário contribui apenas para a popularidade do arquivo i a ele recomendado.

O modelo pode facilmente ser adaptado para o cenário em que os usuários também trazem os arquivos por eles previamente solicitados, adicionando λ_i ao lado direito da equação (3.4), mas nesse trabalho só foi avaliada a situação em que o usuário contribui com o conteúdo só enquanto o consome, descartando depois disso por não ter incentivos para continuar contribuindo com aquele conteúdo.

3.2.3.1 P2P

O custo do modelo desse trabalho representa o custo de disseminação de um conteúdo por um provedor, exatamente conforme MENASCHE et al. (2009), em que, somente foram consideradas as taxas de chegada para cálculo do mesmo. Nesse ambiente, o custo é somente calculado através do custo que um servidor terá para manter o ambiente P2P sem que ele seja auto-sustentável. Para simular esse ambiente, foi assumido que só é gerado custo quando existe somente um *peer* para disseminar o conteúdo. Com isso, o *peer* está atuando como o único servidor que possui o conteúdo.

Para o cálculo exato do custo de disseminação, é aceito que um *swarm* k de um sistema P2P é modelado como um sistema auto-escalável. Em sistemas P2P, arquivos pouco populares dependem do servidor para sua disseminação, enquanto que arquivos muito populares são auto-sustentáveis entre os *peers*, e não requerem sobrecarga no servidor para sua disseminação (vide seção 2.2.1). Sendo μ a capacidade dos *peers* em $Blocos/Tempo$ e a taxa de chegada λ_k em $Chegadas/Tempo$, assume-se, como caso extremo, que o servidor só irá servir *peers* caso haja exatamente um *peer* no *swarm*.

Seja M_k o número de usuários no *swarm* k. A probabilidade de haver exatamente um usuário no *swarm* k, modelado como uma fila $M/G/\infty$, é dada por $P(M_k = 1) = (\lambda_k/\mu)e^{-\lambda_k/\mu}$. O custo para servir cada *swarm* é proporcional a fração de tempo em que o servidor fica ocupado.

$$C_k(\lambda_k) = (\lambda_k/\mu)e^{-\lambda_k/\mu}, \qquad k = 1, \ldots, N \tag{3.5}$$

Conforme a modelagem de uma fila $M/G/\infty$ é considerado que os *peers* es-

tejam chegando ao sistema segundo um fluxo *Poisson* (M) e que sejam servidos de forma arbitrária pelos pelos servidores (G) da rede que, no caso, são infinitos (∞). Ao serem servidos por um único *peer* (servidor) acabam gerando custo de disseminação de conteúdo para o sistema. Essa modelagem foi utilizada para capturar a escalabilidade de sistemas P2P.

Esse custo pode ser adaptado de MENASCHE et al. (2009) introduzindo-se uma unidade monetária ϕ_k que está atrelada ao custo de disseminação de um conteúdo do *swarm* k, sendo que esse valor pode variar dependendo do *swarm*. Para esse trabalho foi adotado $\phi_k = 1$ para todos os *swarms*.

O custo total do sistema, para todos os *swarms* (conteúdos) sem influência do sistema de recomendação é

$$C = \sum_{k=1}^{N} (\lambda_k/\mu)\phi_k e^{-\lambda_k/\mu} \tag{3.6}$$

A figura 3.2 ilustra como que o custo varia em função de λ_k e μ. No restante deste trabalho, foi assumido que $\mu = 1$, sendo assim, simplificando (3.6),

$$C = \sum_{k=1}^{N} \lambda_k e^{-\lambda_k} \tag{3.7}$$

Por sua vez, o custo total perturbado pelo sistema de recomendação é

$$P = \sum_{k=1}^{N} (\gamma_k \mu)\phi_k e^{-\gamma_k/\mu} \tag{3.8}$$

O custo total perturbado é obtido a partir de (3.6), substituindo-se λ_k por γ_k,

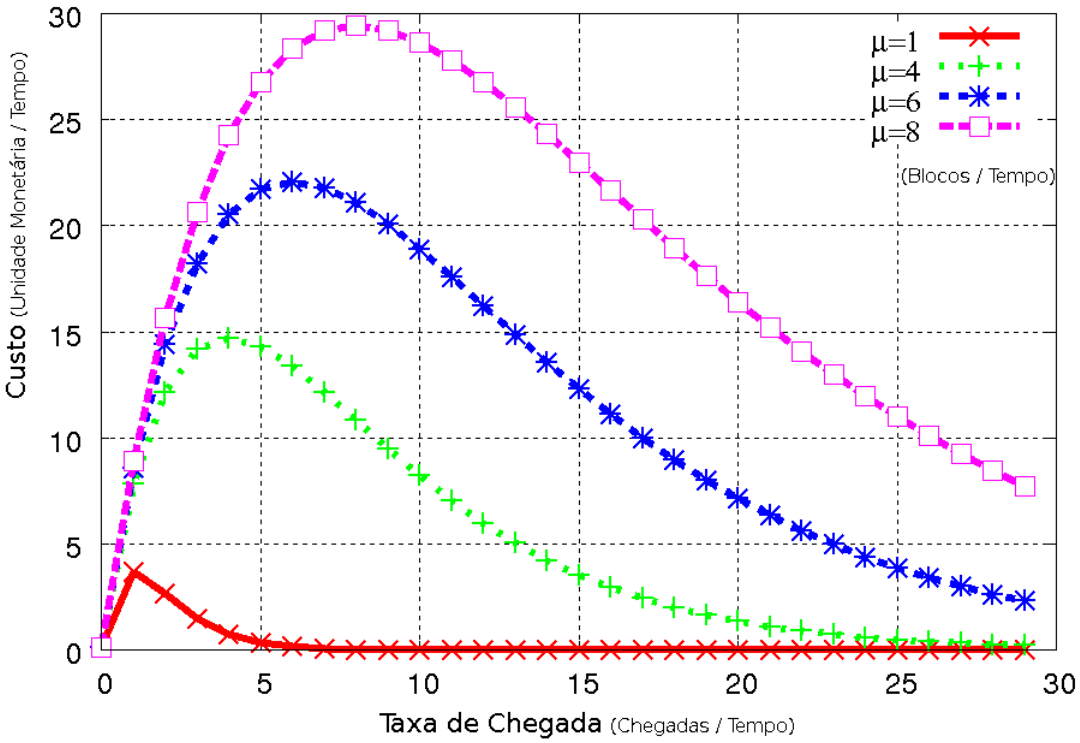


Figura 3.2: Custo de distribuição de um arquivo em função da popularidade do arquivo λ e da capacidade dos peers μ.

vide (3.8). Esta transformação captura o fato de que o SR afeta a taxa de chegada dos usuário ao *swarm* k, que deixa de ser λ_k e passa a ser γ_k, vide (3.4). Novamente é possível simplificar a equação, removendo parâmetros não avaliados (vide (3.9)).

$$P = \sum_{k=1}^{N} \gamma_k e^{-\gamma_k} \tag{3.9}$$

Durante o trabalho foi adotado que um *swarm* possui somente um conteúdo sendo compartilhado, com isso, é possível considerar que $conteúdo_k = swarm_k$ para comparações entre custo de disseminação e recompensa de recomendação.

3.2.3.2 Proxy cache *e ICN*

O modelo desse trabalho é generalizável para outros ambientes de rede que se tenham custo de disseminação e possibilidade de recompensa, como os sistemas de *proxy cache* e redes ICN.

Diferentemente dos sistemas P2P, o custo de disseminação seria determinado pela quantidade de *cache miss*, seja no sistema *proxy cache*, seja em nós de redes ICN.

Para isso, precisa-se definir $M(\lambda_1, \lambda_2, \ldots, \lambda_K)$ como sendo a probabilidade de *cache miss* relativa as respectivas taxas de chegada: λ_1, λ_2, ..., λ_K. Tendo isso, é possível determinar o custo total como:

$$C = M(\lambda_1, \lambda_2, \ldots, \lambda_K) \tag{3.10}$$

E o custo perturbado pelo sistema de recomendação como:

$$P = M(\gamma_1, \gamma_2, \ldots, \gamma_K) \tag{3.11}$$

Durante o trabalho não foram produzidos resultados ou formuladas novas teorias a partir dessa vertente, mas a fim de facilitar trabalhos futuros, foram disponibilizadas ferramentas no Apêndice B e esse *brainstorm* sobre o tema.

3.2.4 Formulação do modelo

O problema consiste em recomendar os melhores conteúdos de acordo com os interesses dos usuários e com o menor custo de disseminação possível. Em nosso modelo, isso corresponde a determinar $p_{k,i}$ ($k = 1, \ldots, N$, e $i = 1, \ldots, N$), de tal forma a minimizar os custos e maximizar as recompensas por recomendação. A seguir, apresentam-se estratégias para lidar com esta tensão entre objetivos conflitantes, usando heurísticas e algoritmos genéticos.

3.3 Heurísticas para recomendação de conteúdo

Nesta seção são propostas heurísticas para a determinação de $p_{k,i}$, $(k,i) \in \{1, \ldots, N\}^2$. Nossas heurísticas são motivadas pelo modelo apresentado na última seção. Sejam $\hat{\lambda}$ e $\hat{r}$ dois limiares (*thresholds*) de popularidade e recompensa. Assume-se que um conteúdo k é popular se $\lambda_k > \hat{\lambda}$ e que a recomendação do conteúdo i para um usuário que requisitou k gera alta recompensa se $r_{k,i} > \hat{r}$. Então, as heurísticas são baseadas nas seguintes duas regras simples

1. **se os usuários que assistem k gostam de i (alta recompensa de recomendação), e i é popular (baixo custo de disseminação), recomendar** i: se $\hat{\lambda} < \lambda_i$ e $\hat{r} < r_{k,i}$, então o conteúdo i tem alta demanda e há alta recompensa em se recomendar o conteúdo i para usuários que requisitam k, logo são favorecidos valores de $p_{k,i}$ mais elevados;

2. **os usuários que assistem k não gostam de i (baixa recompensa de recomendação), e i é popular (já possui baixo custo de disseminação, logo não precisa de *boost*), não recomendar** i: se $\hat{\lambda} < \lambda_i$ e $r_{k,i} < \hat{r}$, então i tem alta demanda (seu custo de disseminação é baixo) e há baixa recompensa

e baixa redução de custos ao se recomendar o conteúdo i para usuários que requisitam k, então são favorecidos valores de $p_{k,i}$ mais moderados;

Dado um conteúdo k, seja n_k^+ o número de conteúdos classificados como de alta prioridade de recomendação (regra 1), n_k^- o número de conteúdos classificados como de baixa prioridade de recomendação (regra 2), e n_k^o os demais, $n_k^- + n_k^+ + n_k^o = N$. Analogamente, dado um conteúdo k, sejam $\mathcal{N}_k^+$, $\mathcal{N}_k^-$ e $\mathcal{N}_k^o$ os respectivos conjuntos de conteúdos de prioridade de recomendação alta, baixa e neutra.

- $\mathcal{N}_k^+ = \{j \in \mathbb{N} \mid \hat{\lambda} < \lambda_j \wedge \hat{r} < r_{kj}\}, \quad n_k^+ = |\mathcal{N}_k^+|$
- $\mathcal{N}_k^- = \{j \in \mathbb{N} \mid \hat{\lambda} < \lambda_j \wedge \hat{r} > r_{kj}\}, \quad n_k^- = |\mathcal{N}_k^-|$
- $\mathcal{N}_k^o = \{j \in \mathbb{N} \mid j \notin \mathcal{N}_k^+ \cup \mathcal{N}_k^-\}, \qquad n_k^o = |\mathcal{N}_k^o|$

A seguir, consideramos três diferentes heurísticas para recomendação de conteúdo construídas usando os conjuntos $\mathcal{N}_k^+$, $\mathcal{N}_k^-$ e $\mathcal{N}_k^o$.

3.3.1 Heurística 1: recomendação uniforme de conteúdos de prioridade alta e neutra

Segundo esta heurística, os conteúdos de prioridade alta e neutra são recomendados de forma aleatória uniformemente. Assim,

$$p_{k,i} = \begin{cases} 1/(n_k^+ + n_k^o), & \text{se } i \in \mathcal{N}_k^+ \cup \mathcal{N}_k^o \\ 0, & \text{caso contrário} \end{cases} \tag{3.12}$$

3.3.2 Heurística 2: recomendação de conteúdos de prioridade alta e neutra, favorecendo os primeiros

Segundo esta heurística, os conteúdos de prioridade alta e neutra são recomendados de forma aleatória uniformemente, mas os primeiros são favorecidos. Para tal, foi introduzido o parâmetro $\epsilon < 1/(n_k^+ + n_k^o)$. Seja ϵ a probabilidade de se redirecionar uma recomendação para um conteúdo de alta prioridade e que, de acordo com a heurística 3.3.1, seria feita para um conteúdo de prioridade neutra. Assim,

$$p_{k,i} = \begin{cases} 1/(n_k^+ + n_k^o) + \epsilon n_k^o / n_k^+, & \text{se } i \in \mathcal{N}_k^+ \\ 1/(n_k^+ + n_k^o) - \epsilon, & \text{se } i \in \mathcal{N}_k^o \\ 0, & \text{caso contrário} \end{cases} \tag{3.13}$$

3.3.3 Heurística 3: recomendação de conteúdos de prioridade alta e neutra, favorecendo os primeiros de acordo com suas recompensas

Esta última heurística é similar à heurística 3.3.2. Entretanto, ao invés de redirecionar-se uniformemente as recomendações que, de acordo com a heurística 3.3.1, seriam feitas para um conteúdo de prioridade neutra, faz-se esse redirecionamento adotando-se as recompensas $r_{k,i}$ como pesos, atribuindo maior peso aos itens que tenham maior $r_{k,i}$. Então,

$$p_{k,i} = \begin{cases} 1/(n_k^+ + n_k^o) + \epsilon n_k^o r_{k,i} / (\sum_{i=1}^{N} r_{k,i}), & \text{se } i \in \mathcal{N}_k^+ \\ 1/(n_k^+ + n_k^o) - \epsilon, & \text{se } i \in \mathcal{N}_k^o \\ 0, & \text{caso contrário} \end{cases} \tag{3.14}$$

3.3.4 Configurar limiares de popularidade e recompensa

Os limiares de popularidade e recompensa são usados para determinar se um determinado conteúdo é popular e se está associado a alta recompensa por recomendações, respectivamente. O limiar $\hat{\lambda}$ é definido como sendo a média das

popularidades,

$$\hat{\lambda} = \frac{\sum_{i=1}^{N} \lambda_i}{N} \tag{3.15}$$

Em seguida, determina-se o limiar de recompensa. Sejam $\overline{r}$ e $\overline{\overline{r}}$ o valor mínimo e máximo de todos os $r_{k,i}$, $\overline{r} = \min_{(k,i)} r_{k,i}$ e $\overline{\overline{r}} = \max_{(k,i)} r_{k,i}$. O limiar de recompensa $\hat{r}$ é definido como a média aritmética entre a recompensa mínima e a máxima,

$$\hat{r} = (\overline{\overline{r}} + \overline{r})/2 \tag{3.16}$$

3.4 Otimização da recomendação utilizando algoritmos genéticos

A seguir, é proposta uma metodologia envolvendo o uso de algoritmos genéticos para buscar o sistema de recomendação de conteúdo ótimo. Em cada cenário, são fixados valores para $r_{k,i}$ e λ_k, $i = 1, \ldots, N$, $k = 1, \ldots, N$, e é executado o algoritmo genético para determinação dos $p_{k,i}$ ótimos. Esses $p_{k,i}$ são otimizados através de um vetor com a quantidade de $p_{k,i}$ (genes) que devem ser otimizados (vide Figura 3.3). São testados vários valores de $p_{k,i}$ para cada gene do vetor, encontrando heuristicamente os $p_{k,i}$ mais apropriados para cada cenário. O programa MatLab® 2009 com a extensão de algoritmo genético (AG) foi utilizado para criação do algoritmo e execução do mesmo.

O nível de *fitness* é dado pela recompensa total menos o custo total normalizado. O custo total normalizado $\overline{C}$ é determinado de tal forma a aumentar a penalização onde o custo perturbado (P) esteja muito distante do custo de referência (C). Esta ideia é inspirada na maximização da recompensa sob a restrição de que o custo não ultrapasse significativamente o custo de referência.

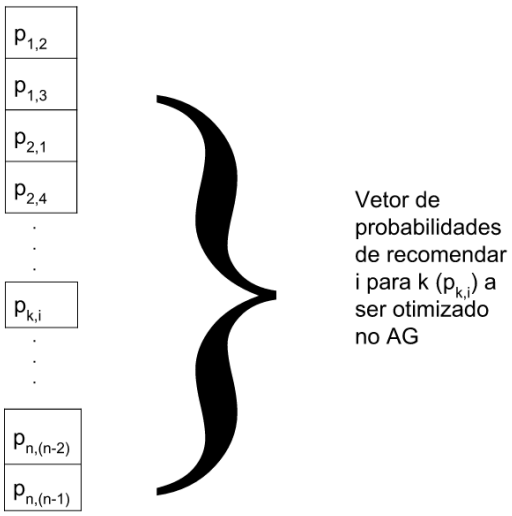


Figura 3.3: Cada $p_{k,i}$ é um gene no vetor do Algoritmo Genético (AG) utilizado em todos os experimentos desse trabalho. Cada vetor do AG possui combinações diferentes de genes, sendo assim, várias combinações de $p_{k,i}$ são testadas para obter o melhor resultado.

Além disso, o custo normalizado simplifica o balanceamento da diferença entre os custos perturbado (P) e de referência (C) com a recompensa (R) na função de *fitness* 3.18.

Neste trabalho, considera-se uma função degrau para o custo normalizado que penaliza mais fortemente custos que sejam muito acima do valor de referência, sendo que no primeiro degrau captura sempre o valor mais baixo de custo para maior otimização da função de *fitness* como indicado na equação (3.17).

$$\overline{C} = \begin{cases} \text{P - C}, & \text{se } P - C < -1 \\ 10.000, & \text{se } -1 \leq P - C < 1 \\ 1.000.000, & \text{se } 1 \leq P - C < 10 \\ 100.000.000, & \text{caso contrário} \end{cases} \tag{3.17}$$

Sendo assim, $\overline{C}$ é uma função conversora de $Unidade\ Monetária$ (C e P) para $PontosLikert$, permitindo a comparação entre R e o resultado de $\overline{C}$ para gerar o valor de *fitness* em $\frac{Pontos\ Likert}{Tempo}$. A função de *fitness* F a ser maximizada é a

recompensa R dada por (3.3) menos o custo normalizado dado por (3.17),

$$F = R - \overline{C} \tag{3.18}$$

Nos resultados experimentais, o objetivo é maximizar (3.18) para obter os maiores valores de recompensa com os menores valores de custo.

3.4.1 Cenário com dois arquivos

Na primeira análise via algoritmos genéticos, considera-se em cada cenário apenas dois arquivos, A e B, para estudar a interação dos $p_{k,i}$ com o AG e os parâmetros que necessitam de mais atenção para obtenção dos $p_{k,i}$ desejados. Com isso, são obtidos resultados no ambiente mais simples possível para depois ampliar a base de estudo. Dados um perfil de recompensas e popularidades, ou seja, dadas as recompensas $r_{A,B}, r_{B,A}$ e as taxas de chegada λ_A e λ_B, o objetivo é obter $p_{A,B}$ e $p_{B,A}$. Para tal, considera-se uma população de 200 indivíduos com dois genes $p^{(l)}_{A,B}$ e $p^{(l)}_{B,A}$, $A, B \in \{1, \ldots, N\}$, em cada indivíduo l da população, $l = 1, \ldots, 200$. Foi avaliada então a evolução da população por 50 gerações e utilizados os seguintes parâmetros no AG: a) 2 indivíduos em elitismo; b) a seleção é proporcional utilizando o método do torneio entre os 4 melhores indivíduos; c) *crossover* em um ponto único com taxa de mutação de 1% (vide Figura 3.4).

Note que, enquanto nas seções anteriores foi assumido que $\sum_{i=1}^{N} p_{k,i} = 1$, nessa seção considera-se o caso em que $\sum_{i=1}^{N} p_{k,i} \leq 1$. Dessa forma, usuários que não receberem recomendação de conteúdo vão embora do sistema imediatamente após consumirem o conteúdo de interesse. Isto ocorre, por exemplo, se a recomendação de certo conteúdo geraria custo tão alto que é vantajoso não recomendar ao usuário

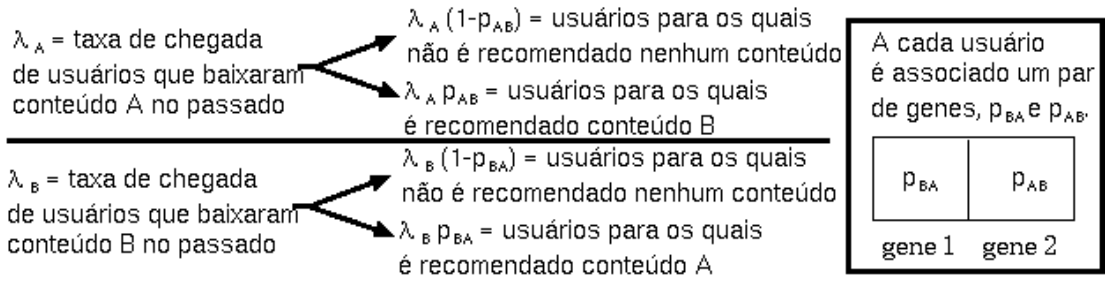


Figura 3.4: Na análise via algoritmos genéticos mais simples, consideram-se apenas dois arquivos. O objetivo é obter $p_{A,B}$ e $p_{B,A}$.

nenhum conteúdo.

Após executar-se o algoritmo genético por uma geração, obtêm-se um novo conjunto de amostras de $p_{A,B}^{(l)}$ e $p_{B,A}^{(l)}$, para $l = 1, \ldots, 200$. Dado o perfil de popularidades e recompensas, obtêm-se o custo perturbado pelo SR, P, o custo de referência, C e a recompensa total, R, dados por (3.9), (3.7) e (3.3), respectivamente. O nível de *fitness* da população é dado por (3.18). Repete-se o algoritmo por 50 gerações.

3.4.2 *Dataset* do MovieLens

O intuito agora é usar o AG para analisar o *trace* do MovieLens, a fim de encontrar os valores de $p_{k,i}$ que maximizem a função de *fitness* e comparar o custo e a recompensa resultantes com aquelas obtidas usando as heurísticas propostas. Para isso, foram utilizados os mesmos parâmetros da seção 3.4, envolvendo 551 filmes amostrados do *trace* de 1,000 avaliações do MovieLens. Note que temos agora 87 genes a serem otimizados, correspondendo aos 87 valores de $p_{k,i}$ para os quais $r_{k,i} > 0$ (assume-se que $r_{k,i} = 0 \Rightarrow p_{k,i} = 0$). O algoritmo foi repetido por 100 gerações.

Para determinar o melhor equilíbrio entre a função de custo e de recomenda-

ção no sistema (3.18) foi utilizada a mesma configuração do AG para proporções de custo ($\overline{C}$) e recompensa (R) como $\{(R\times 0, \overline{C}\times 1), (R\times 0.1, \overline{C}\times 0.9) \ldots (R\times 1, \overline{C}\times 0)\}$. Nesse experimento foram utilizadas somente 20 gerações, pois no primeiro foi observada uma rápida convergência do algoritmo.

Durante todos os experimentos com o MovieLens foi definido que $\sum_{i=1}^{N} p_{k,i} = 1$, então foi necessário criar novas funções para: criação da população de 200 indivíduos; cruzamento de indivíduos (*crossover*); e restrição em caso de resultar em indivíduos inválidos durante a mutação.

Todos os experimentos aqui elaborados visam determinar as melhores heurísticas de recomendação de conteúdo para o modelo proposto utilizando a base de dados do MovieLens, sendo que essa recomendação influencia diretamente na probabilidade do usuário ir do conteúdo k para o conteúdo i ($p_{k,i}$). Como a base do MovieLens possui avaliações sequenciais, já é possível conhecer as preferências de alguns usuários com relação a ir de um conteúdo r para um i ($r_{k,i}$). Além das heurísticas conceituais, foram feitas heurísticas através de otimizações com AG para melhorar essas heurísticas conceituais, pois o AG se torna muito custoso de implementar em qualquer ambiente de rede e só funciona com soluções aproximadas. Nos próximos capítulos serão mostrados os resultados do planejamento descrito nesse capítulo.

4 RESULTADOS

Nesse capítulo serão comentados todos os resultados obtidos para análise e formulação do modelo matemático proposto para juntar SR e QoE (aqui representado somente pelo custo de disseminação). Como primeiros resultados, é verificada a relevância de recomendação no ambiente de estudo usando um simulador P2P.

Depois de compreender um pouco mais o sistema e a importância do SR, o modelo matemático foi destrinchado para estudar a relação entre SR e custo de disseminação de conteúdo. Esse modelo foi avaliado primeiro em um simulador com as heurísticas propostas e, posteriormente, utilizado em várias configurações de algoritmo genético (AG) para buscar os melhores parâmetros que podem equilibrar custo de disseminação de conteúdo em redes P2P e preferência teórica dos usuários, ou seja, sem estudo com usuários reais.

4.1 Relevância da recomendação para sistemas P2P

Para análise da rede P2P e as implicações que a recomendação de conteúdo pode provocar nessas redes foi implementado em Python um simulador com diversas configurações visando verificar *downloads* de conteúdo via P2P e consumo desses conteúdos. Cada *peer* participante do sistema possui uma lista de conteúdos que já traz consigo e outra lista de conteúdos que irá consumir, chamada de (*wishlist*). Assume-se que a interseção dos conteúdos que o usuário traz consigo ao sistema e de sua *wishlist* é vazia. Como parâmetros da simulação, foram utilizados 50 *peers*, sendo que cada um deles possuía uma *wishlist* e uma lista de conteúdos com 120 arquivos cada. Essas duas listas derivavam sempre do total de 5000 conteúdos do

sistema de forma aleatória. Esses conteúdos são somente compostos pelo nome do conteúdo e por um valor unitário para o tamanho do conteúdo.

O tempo foi dividido em *timeslots* e os *peers* interagiam a cada *timeslot* pareando aleatoriamente e verificando se estavam interessados em fazer *download* de algum conteúdo que o *peer* com quem pareararam possuía em sua lista de conteúdos. Na simulação foram mantidos todos os parâmetros fixos exceto a taxa de *download*, o tipo de *download* e o tipo de consumo.

O *download* ocorre quando um *peer* pega um arquivo da lista de conteúdos do outro *peer* para sua lista de conteúdos de acordo com sua taxa de *download*. As taxas de download utilizadas foram 10%, 30%, 50%, 70% e 100% e são definidas como a porcentagem do conteúdo a ser feito o *download* a cada *timeslot*. O tipo de *download* pode ser sequencial, em que os *peers* realizam *download* dos conteúdos da *wishlist* sequencialmente; ou aleatório, em que o *peer* efetua *download* de qualquer conteúdo da sua *wishlist* que esteja na lista de conteúdos do *peer* com quem pareou. O *download* aleatório simula um SR que indica um melhor conteúdo a ser feito *download* dentro das preferências do usuário (*wishlist*). Cada *download* é feito totalmente ou parcialmente dependendo da taxa de download estipulada.

Depois de realizar um *download*, os *peers* consomem os conteúdos da *wishlist* que sofreram *download* e o *timeslot* se encerra. O consumo ocorre segundo uma taxa de consumo. A taxa de consumo utilizada foi fixada em 30% e é a porcentagem do arquivo a ser consumido a cada *timeslot*. Esse consumo é o equivalente a ouvir um áudio ou assistir um vídeo em um sistema AVoD, então assume-se que o *peer* só começa a consumir outro conteúdo quando termina o corrente. O tipo de consumo pode ser sequencial em que os *peers* realizam consumo dos conteúdos da *wishlist* sequencialmente; ou aleatório em que a ordem de consumo é escolhida de forma aleatória entre conteúdos com *download* concluídos ou parcialmente concluí-

dos, possibilitando uma segunda camada de abstração para o SR.

Em cada simulação, todos os *peers* possuem as taxas de *download* e consumo fixas em um dos valores pré-definidos. Cada rodada de simulação dura 120 *timeslots* e foram repetidas 10 rodadas para cada grupo de parâmetros. A Figura 4.1 ilustra o funcionamento do simulador P2P.

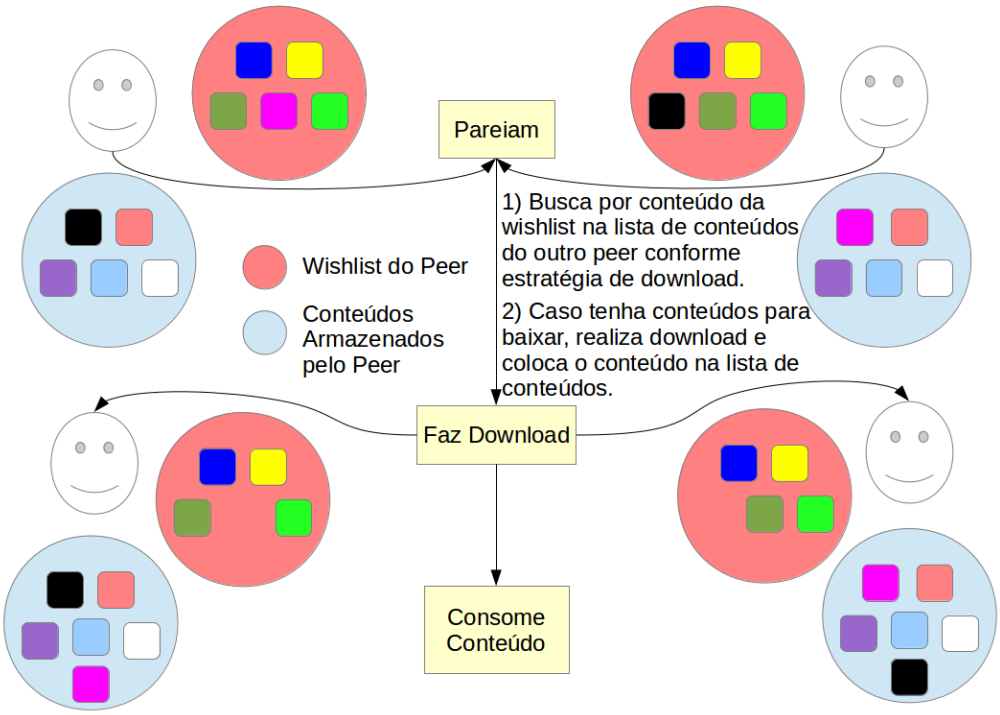


Figura 4.1: Ilustração de funcionamento do simulador P2P. Os *peers* pareiam aleatoriamente e buscam por conteúdos de sua *wishlist* conforme sua estratégia de *download*. Caso o *download* seja feito o conteúdo é adicionado a sua lista de conteúdos para ser consumido. O consumo é realizado caso haja arquivo na lista de conteúdos para ser consumido de acordo com sua estratégia de consumo.

Como saída, o simulador gera gráficos de:

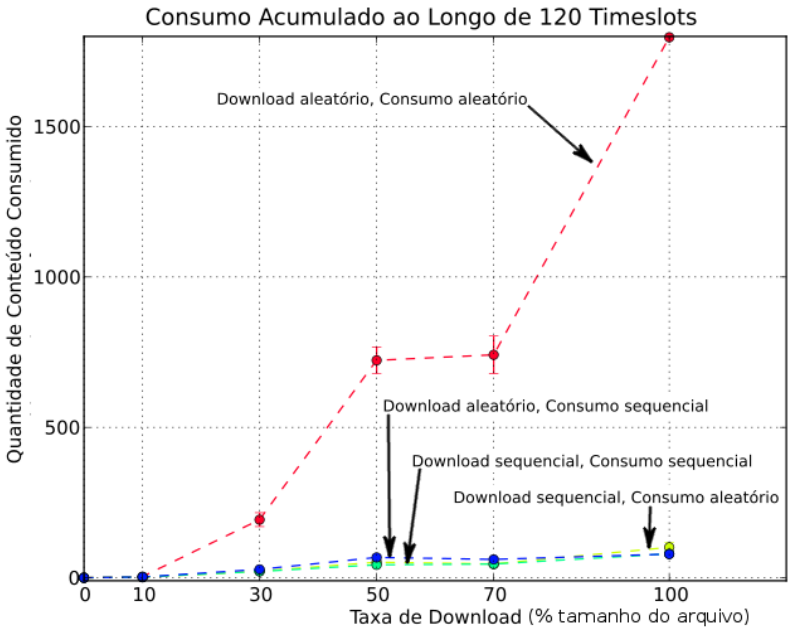
- simulação *versus* quantidade de downloads concluídos no total;

- simulação *versus* quantidade de consumos concluídos no total;
- simulação *versus* quantidade de *peers* sem download no total;
- simulação *versus* quantidade de *peers* sem consumo no total;
- simulação *versus* parcela de pareamentos que resultaram em *download* e parcela que não resultaram em *download*.

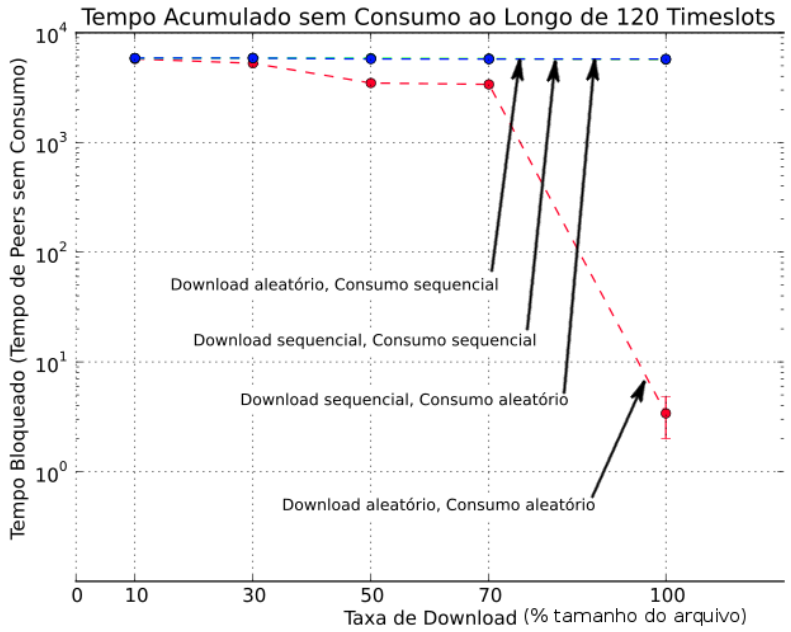
O objetivo da construção desse sistema é verificar como um SR conseguiria controlar tanto a preferência dos usuários, quanto a dificuldade com que os usuários recebem e consomem conteúdo.

Com esse objetivo, são apresentados os resultados referentes a cenários em que os *peers* não têm flexibilidade para ajustar seus *downloads* e consumos de acordo com recomendações do sistema. Nesses casos, os dois principais desafios encontrados em nossas simulações foram: 1) eventualmente *peers* ficam bloqueados sem ter conteúdo para consumir e 2) ao final do tempo de simulação muitos *peers* não conseguem obter grande parte do conteúdo desejado. Então, caso seja assumido que os *peers* têm flexibilidade para ajustar seus *downloads* e consumos, a situação muda radicalmente.

Nas Figuras 4.2a e 4.2b apresenta-se o consumo acumulado de conteúdos de todos os *peers* e o tempo acumulado sem consumo de todos os *peers* em *timeslots*, respectivamente, em função da taxa de *download*. Essas duas figuras servem para ilustrar os problemas 1 e 2 descritos acima, respectivamente. Na Figura 4.2a percebe-se que as simulações com *consumo sequencial*, independente da taxa de *download* ou do tipo de *download* (sequencial ou aleatório), resultam em muito tempo com os *peers* bloqueados. O mesmo se repete com as simulações com *download sequencial - consumo aleatório*, já que o *download* força o consumo a ser sequencial também,



(a) Quantidade de conteúdos completamente consumidos pelos peers em escala logarítmica.



(b) Tempo que peers ficam bloqueados em escala logarítmica.

Figura 4.2: Resultado do consumo dos peers para simulações realizadas. (a) Observa-se que os ganhos obtidos quando se combina flexibilidade no consumo com flexibilidade no download são grandes. (b) Somente com download e consumo aleatórios foi possível consumo contínuo.

pois não são feitos *downloads* fora da ordem da *wishlist*. A Figura 4.2b ilustra outro problema dos sistemas com *download sequencial - consumo aleatório*, *download sequencial - consumo sequencial* e *download aleatório - consumo sequencial*. Nesses sistemas, os *peers* não conseguem fazer *download* de todo o conteúdo desejado. A quantidade de conteúdos consumidos cresce sutilmente na medida em que a taxa de *download* aumenta, mas não passou de 90 conteúdos ao longo dos 120 *timeslots*, em nenhuma das rodadas de simulação consideradas.

As Figuras 4.2a e 4.2b mostram que o comportamento do sistema muda radicalmente quando considera-se que os peers têm flexibilidade para decidir a ordem de *download* e consumo do conteúdo (curvas vermelhas). Nesse caso (*download* e consumo aleatórios), isso ocorre devido ao aumento da probabilidade de *download* de um conteúdo durante o pareamento entre dois *peers* por pelo menos um desses *peers*, pois a janela de busca aumenta. Além disso, o consumo aleatório permite que qualquer arquivo da *wishlist*, que tenha seu *download* em andamento ou concluído, seja consumido parcialmente ou completamente, respectivamente.

Note que é essencial a flexibilidade tanto do *download* quanto do consumo. Se assumir que apenas o consumo é flexível (curva *download sequencial - consumo aleatório*), o desempenho do sistema não é satisfatório. Dessa forma, é fundamental que qualquer sistema de recomendação que vise aumentar o QoE afete não apenas o consumo, mas também a ordem em que os conteúdos são baixados. [1]

[1] Mais detalhes e o código fonte do simulador P2P estão disponíveis em: `http://code.google.com/p/simuqos`

4.2 Heurísticas

Nesta seção, são apresentados os resultados obtidos com as heurísticas da seção 3.3. Como referência, considera-se o conjunto de $N = 551$ arquivos introduzido na seção 3.1. Tanto a taxa de chegada dos arquivos quanto as recompensas são definidas usando um *trace* do MovieLens (MOVIELENS, 2013), de acordo com a metodologia descrita na seção 3.1.

A seguir, comparam-se os custos e recompensas de referência *versus* aqueles obtidos com as heurísticas propostas na seção 3.3. As curvas na Figura 4.3 ilustram como os custos e recompensas variam em função do somatório das taxas de chegada Λ. Nessas curvas foi considerado Λ variando de 1 até 20930, em incrementos de 70 (vide eq. (3.1)).

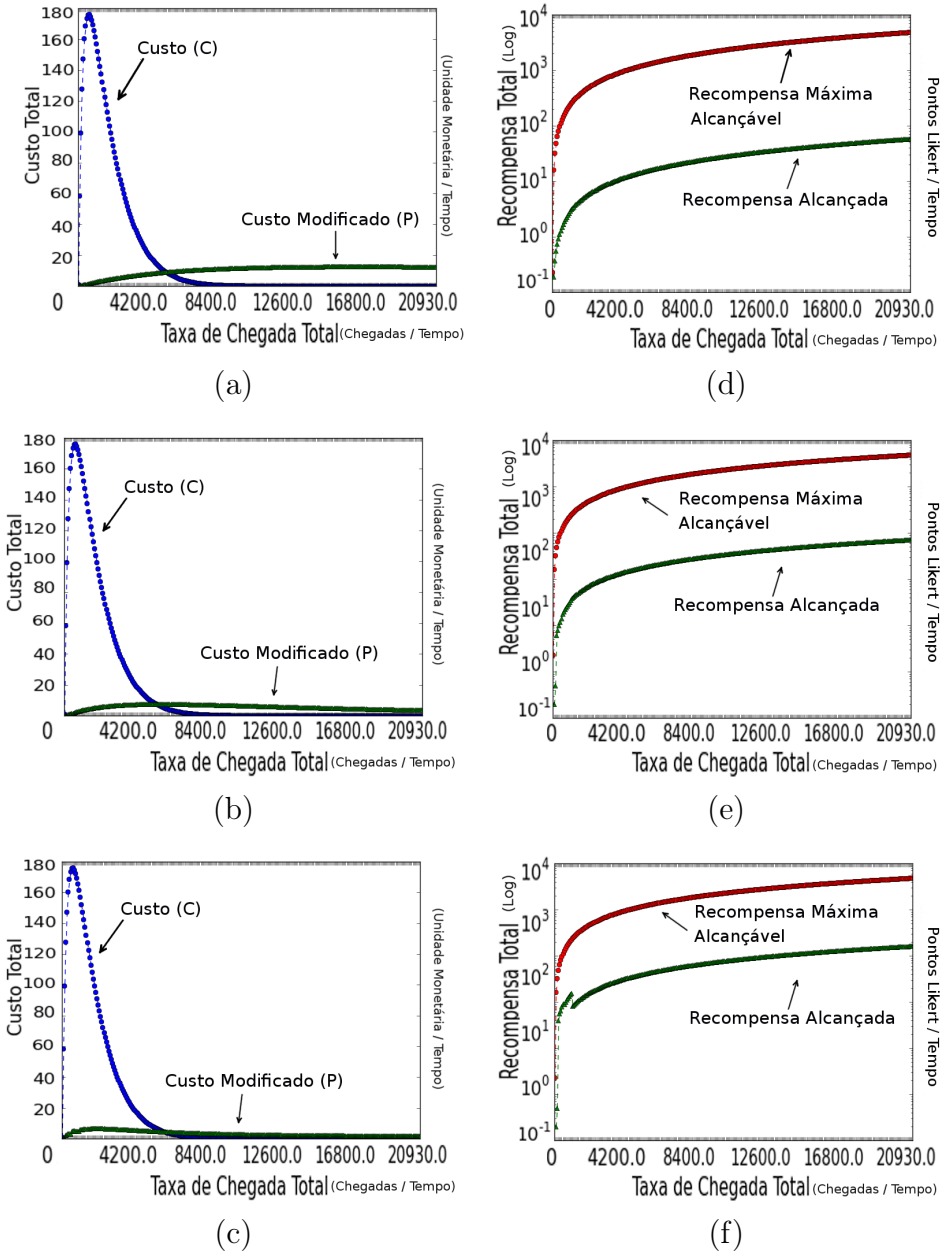


Figura 4.3: Tradeoffs envolvidos no sistema de recomendação (custos e recompensas). Resultados com heurísticas 1, 2 e 3, nas linhas 1, 2 e 3, respectivamente. Na primeira e segunda colunas estão as curvas de custos (sendo **C**, sem sistema de recomendação e **P** com sistema de recomendação) e recompensas (mais em cima (em vermelho), máximo alcançável, com sistema de recomendação que não leva em conta a rede, e mais embaixo (em verde), valor obtido com recomendação que leva em conta a rede). As heurísticas apresentaram padrões de custos similares, sendo que a terceira destacou-se pela sua maior recompensa e menor custo. Recompensa está em escala logarítmica.

As curvas verdes (**P**) de custos (Figuras 4.3(a)-(c)) correspondem aos valores obtidos com o sistema de recomendação, e as azuis sem ele. Note que, ao utilizar o sistema de recomendação, o pico do custo diminui significativamente para qualquer uma das heurísticas adotadas.

Nos gráficos de recompensas (Figuras 4.3(d)-(f)), as curvas vermelhas (menores valores) correspondem à recompensas obtidas caso o sistema de recomendação não levasse em conta os fatores relativos à rede. Nesse cenário clássico, recomenda-se para o usuário que assistiu k o conteúdo i tal que $r_{k,i}$ seja maximizado, ou seja, aumentando a satisfação ($r_{k,i}$) do usuário que assistiu k e agora pode assistir i. As Figuras 4.3(a)-(c) quantificam a diminuição do custo ao adotar-se o sistema de recomendação proposto (**P**) contra o cenário clássico (**C**). Note que a heurística 3 gerou recompensas maiores e custos menores que as heurísticas 1 e 2, sendo assim considerada superior no cenário em questão.

4.3 Algoritmo genético

A seguir, é analisado o comportamento do algoritmo genético em um cenário simplificado com dois arquivos (Seção 4.3.1) e usando o trace do MovieLens (Seção 4.3.2). No primeiro caso serviu para uma análise qualitativa de comparação das heurísticas com os resultados obtidos via AG, verificando em que situações as heurísticas são corroboradas pelos resultados obtidos via AG, enquanto que no segundo caso foi feita uma comparação quantitativa dos resultados obtidos com as heurísticas na seção anterior contra aqueles obtidos usando o AG.

4.3.1 Cenário com dois arquivos

A seguir, são apresentados os resultados obtidos com o algoritmo genético proposto na seção 3.4. Consideramos um cenário de 2 arquivos, A e B. Para esses arquivos, o algoritmo genético tenta otimizar os valores de $p_{A,B}$ e $p_{B,A}$ a cada rodada, buscando em uma população de 200 indivíduos. Cada indivíduo possui inicialmente um $p_{A,B}$ e um $p_{B,A}$ escolhidos de forma uniforme e aleatória de 0 a 1, sendo que esses parâmetros $p_{A,B}$ e $p_{B,A}$ associados a cada indivíduo são chamados de **genes**. A intenção do AG é confrontar esses 200 indivíduos e encontrar as melhores combinações de $p_{A,B}$ e $p_{B,A}$ que tenham um custo total baixo e uma recompensa total alta.

Os custos totais de disseminação dos arquivos, originais e perturbados pelo mecanismo de recomendação, são dados pelas equações (3.7) e (3.9), respectivamente. Às recompensas $r_{A,B}$ e $r_{B,A}$ são atribuídos valores 0,1 (baixa), 0,5 (média) ou 0,9 (alta), e às taxas de chegada λ_A e λ_B são atribuidos valores de 5 a 30, de acordo com os objetivos do experimento em questão. Os cenários analisados são listados na Tabela 4.1. Nela, aparecem quatro grandes grupos de parâmetros que foram categorizados como:

- *Grupo de parâmetros G1:* conteúdos com alta recompensa e alta taxa de chegada;
- *Grupo de parâmetros G2:* conteúdos com baixa recompensa e alta taxa de chegada;
- *Grupo de parâmetros G3:* conteúdos com baixa recompensa e baixa taxa de chegada;
- *Grupo de parâmetros G4:* conteúdos com taxa de chegada intermediária, esti-

pulada como 15.

cenário	$r_{A,B}$	$r_{B,A}$	λ_A	λ_B	$p_{A,B}$	$p_{B,A}$	*score*
Grupo G1: Conteúdo com alta recompensa e alta taxa de chegada							
1	0,9	0,9	30	30	0,993292	0,998423	-9704
Grupo G2: Conteúdo com baixa recompensa e alta taxa de chegada							
2	0,1	0,1	30	30	0,985521	0,972787	-9967
Grupo G3: Conteúdo com baixa recompensa e baixa taxa de chegada							
3	0,1	0,1	5	5	0,01	0,01	-9999
Grupo G4: Conteúdo com λ intermediário estipulado como 15.							
4	0,1	0,1	15	15	0,990565	0,988124	-999983
5	0,5	0,5	15	15	0,995851	0,99	-999918
6	0,9	0,9	15	15	0,993053	0,994235	-999852

Tabela 4.1: Alguns resultados obtidos com o AG. A tabela está dividida respectivamente entre os quatro grupos de parâmetros estudados. Cada grupo possui subgrupo(s) com uma taxa de chegada λ, um $r_{k,i}$ e um $p_{k,i}$ para os arquivos A e B, além do melhor *score* (combinação entre custo e recompensa) encontrado pelo AG.

No grupo G1 da Tabela 4.1, é possível visualizar que os $p_{A,B}$ e $p_{B,A}$ estão bem próximos de 1. Sendo assim, o AG está informando que é interessante recomendar esses arquivos, por terem grande quantidade de usuários interessados neles (alto λ) e por serem muito desejados por esses usuários (alto $r_{A,B}$ e $r_{B,A}$). Esse grupo corrobora com a regra 1 da seção 3.3 para criação das heurísticas de $p_{k,i}$: se $\hat{\lambda} < \lambda_i$ e $\hat{r} < r_{k,i}$, então recebe $p_{k,i}$ mais elevado.

Já no grupo G2, o AG aparentemente determinou uma solução que vai de encontro ao estabelecido na regra 2 da seção 3.3: se $\hat{\lambda} < \lambda_i$ e $r_{k,i} < \hat{r}$, então recebe $p_{k,i}$ mais moderados. Nesse grupo, existem dois arquivos muito populares (alto λ), mas não recomendáveis um ao outro (baixo $r_{A,B}$ e $r_{B,A}$). Mesmo assim, o AG determinou altos valores de $p_{A,B}$ e $p_{B,A}$ (próximos de 1). Isso pode ser explicado pelo fato de λ muito alto estar contribuindo muito mais na função de custo do que o $r_{k,i}$ baixo na recompensa. Por ter um λ muito elevado, naturalmente as funções de custo das equações (3.5) e (3.9) tendem a diminuir muito ao recomendarmos os

arquivos, mesmo que eles não sejam fortemente complementares. Assim, na ausência de outras possibilidades o AG favorece a recomendação em prol da diminuição dos custos de rede.

No grupo G3 da Tabela 4.1, o AG corrobora com a regra 2 das heurísticas. Para um conteúdo pouco popular (baixo λ) e com pouco interesse via recomendação por parte dos usuários (baixo $r_{A,B}$ e $r_{B,A}$), não parece ser interessante recomendar o conteúdo para qualquer usuário. Com isso, ficou estabelecido pelo AG que conteúdos desse tipo receberiam valores próximos a 0 de $p_{A,B}$ e $p_{B,A}$.

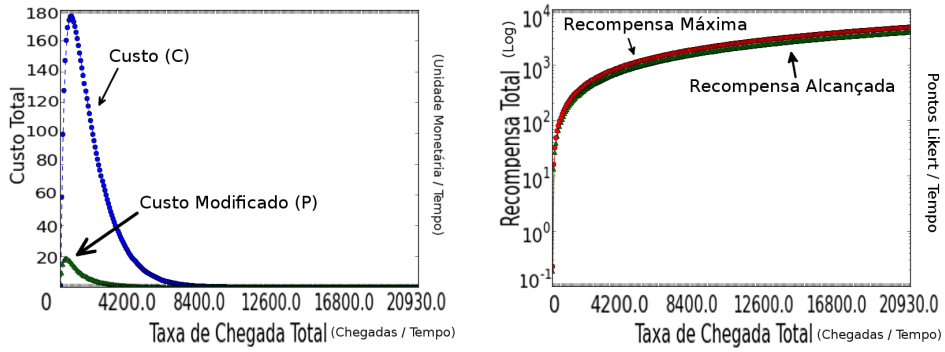
No grupo G4 da Tabela 4.1, com $\lambda = 15$, foram considerados três subgrupos. Para os três subgrupos, o *score* é menor que o dos grupos 1, 2 e 3. Quando $\lambda = 15$, os custos são muito elevados tornando a recomendação muito custosa. Isto dificulta o AG de determinar os parâmetros ótimos neste grupo. Trabalhos futuros consistem em avaliar políticas alternativas para estas situações.

A partir de uma análise detalhada de todos os grupos de parâmetros é possível determinar valores adequados de $p_{k,i}$ para diferentes ecossistemas de variáveis. Isso simplifica a criação de um SR que leve em conta as características de cada grupo de parâmetros e, assim, pré-estabelecer valores de $p_{k,i}$ adequados para diferentes situações.

4.3.2 Inferindo $p_{k,i}$ de MovieLens com algoritmo genético

Depois de explorar um ambiente mais simples com dois arquivos, foram obtidos alguns resultados com o *trace* do MovieLens e o AG. A Figura 4.4, ilustra os resultados do AG que podem ser comparados com as heurísticas da Figura 4.3. Em (a) e (b) é possível verificar os resultados de custo e recompensa, respectiva-

mente, para os $p_{k,i}$ determinados pelo AG. O AG conseguiu diminuir o custo total do sistema e aumentar a recompensa até bem perto do valor máximo de recompensa total. Esses resultados ilustraram maior recompensa e menor custo que os resultados obtidos via melhor heurística (heurística 3, vide Seção 3.3.3).



(a) (b)

Figura 4.4: Tradeoffs envolvidos no sistema de recomendação com AG (custos e recompensas). Os $p_{k,i}$ gerados pelo AG reduziram o custo e aumentaram a recompensa comparados a Figura 4.3. Recompensa novamente está em escala logarítmica.

4.3.3 Analisando $p_{k,i}$ gerados por algoritmo genético

Ao verificar que os valores de $p_{k,i}$ foram melhor determinados através de algoritmos genéticos, fez-se necessário analisar como o AG chegou a esses valores de $p_{k,i}$. Como a técnica de AG não é determinística, foram feitas rodadas com combinações de R e C para valor final de *fitness* (eq. (3.18)). As combinações para proporções de custo ($\overline{C}$) e recompensa (R) foram descritas na seção 3.4.2 e os resultados expostos no Apêndice C levaram as seguintes deduções:

1. mesmo com λ_k baixo, quando $r_{k,i}$ é alto o AG determinou alto $p_{k,i}$;
2. quando $\sum_{k=1}^{N} r_{k,i}$ é dividido entre todos os k o AG determinou $p_{k,i}$ alto para somente um dos k;

3. quando λ_k é alto o AG determinou alto $p_{k,i}$;

4. quando $\sum_{k=1}^{N} r_{k,i}$ é dividido entre todos os k e um dos k possui λ_k mais alto o AG determinou $p_{k,i}$ alto para somente para aquele k;

5. quando todos os $r_{k,i}$ de um i são baixos, um deles recebe alto $p_{k,i}$ enquanto os outros recebem valores baixos. Isso ocorre devido ao fator de atenuação $\beta = 1$ da equação 3.2.

6. quando todos os $r_{k,i}$ e os λ_k de um i são baixos, o AG não teve uma posição definida e deu valores aleatórios aos $p_{k,i}$.

As deduções 1 e 3 respeitaram exatamente a regra 1 da seção 3.3, determinando $p_{k,i}$ alto para essas duas situações e sempre priorizando o $r_{k,i}$ alto. Na dedução 4, foi priorizado o λ_k mais alto, para influenciar a diminuição do custo já que todos os conteúdos são igualmente bons (mesmo $r_{k,i}$ para todos os k). Para as deduções 5 e 6 o AG deve encontrar uma solução melhor ao variar o valor do fator de atenuação β, pois a variação desse fator vai possibilitar que o usuário do modelo continue ou não vendo e assistindo conteúdos conforme descrito na seção 3.1. A dedução 2 é uma decisão totalmente voltada para a diminuição do custo por parte do AG, pois caso não seja impactado pela dedução 4, não existe diferença entre os $r_{k,i}$ e nem entre os λ_k, então o AG tenta aumentar um dos λ_k aleatoriamente.

5 CONCLUSÃO

O crescente aumento de indivíduos que possuem acesso a internet e o aumento de dispositivos com rede 3/4G, reflete diretamente na quantidade de acessos a conteúdos *WEB*, P2P, etc. Conforme descrito nesse trabalho, percebe-se a possibilidade de implantar um SR que facilite a disseminação de conteúdo através da preferência dos usuários.

Uma disseminação de conteúdo mais eficiente pode se tornar bastante interessante por fins econômicos, ao conseguir maior satisfação dos usuários de um determinado serviço, ou simplesmente para economizar recursos para expansão desse serviço. Assim, um indivíduo poderia conseguir um conteúdo com maior velocidade, podendo melhorar a QoE oferecida por determinado serviço a esse indivíduo. Isso segue exatamente os princípios de HCM (Multimídia Centrado nos Humanos), mas através de uma camada mais externa de abstração, não considerando pedaços de um conteúdos, mas sim o conteúdo inteiro.

Por ser um trabalho em uma nova linha de pesquisa, nessa dissertação foram apresentados muitos resultados inovadores e, inclusive, possibilidades de novas linhas de pesquisa.

Nesse trabalho, foi bem definido um modelo matemático que junte recomendação de conteúdo e custo de disseminação (sessão 3.2). Nesse modelo foi identificado $p_{k,i}$, uma variável de controle simples de se trabalhar e que permite o controle de QoE do sistema tanto para administradores de rede como para usuários finais (sessão 3.2.1.2). Além disso, o modelo em questão já teve vários parâmetros identificados para sua otimização, sendo eles: ϵ, β, q, ϕ e μ.

Em cima desse modelo foram criadas 3 heurísticas que foram bem avaliadas para o cenário mais simplista (sessão 4.2), mas que precisam ser testadas em outros cenários mais complexos e com a variação dos parâmetros não estudados do modelo. Devido ao acoplamento simples do $p_{k,i}$ no modelo, é possível criar novas heurísticas ou definir esse $p_{k,i}$ através de cálculos como multiplicadores de Lagrange.

Como primeira contribuição para área de Sistemas de Recomendação, foi ilustrado nesse trabalho a importância desses sistemas para diminuição do custo de disseminação em redes P2P (sessão 4.1). Outra contribuição importante foi na forma de medir a eficiência de recomendações através da recompensa (vide equações (3.2) e (3.3)) em sistemas que não seja possível medir precisão e revocação (sessão 3.1).

Por último na área de recomendação de conteúdo, foi analisado que para criação de um SR real que se preocupe em diminuir o custo de disseminação é necessário que esse sistema gerencie tanto a ordem com que o conteúdo é adquirido, como também a ordem com que é consumido (sessão 4.1), pois somente assim é possível conseguir o menor custo com a maior satisfação do usuário.

Ao combinar otimização e redes também existiram contribuições, visto que foi produzido um modelo quase ótimo de Algoritmo Genético juntando SR e disseminação de conteúdo (sessão 3.4). Os resultados com AG foram bastante superiores aos das heurísticas, mas sempre é mais complicado implementar AG dentro de um sistema, além de ter resultados mais demorados que as heurísticas. A demora na obtenção de resultados não inviabiliza a utilização do modelo com AG, visto que o poder computacional tem aumentado cada vez mais e o AG pode realizar seus cálculos enquanto o usuário consome o último conteúdo recomendado.

Ainda sobre os AGs, esse trabalho identificou os grupos de parâmetros que devem ter mais atenção ao utilizar a otimização (sessão 4.3.1). Além de detalhar

como esses algoritmos estão obtendo seus resultados, também foram apontadas soluções para melhorias das heurísticas (sessão 4.3.3) atuais e outras que possam ser criadas.

Para implementação de trabalhos futuros, foram deixadas algumas dicas pelas descobertas aqui realizadas. Dentre elas, estão o *brainstorm* sobre *cache* e redes orientadas a conteúdo, além das ferramentas abertas para continuação tanto do trabalho para redes P2P, como *cache* e rede de *caches* (sessões 4.1 e 3.2.3.2 e Apêndice B).

Com as análises feitas nesse trabalho é possível produzir novas heurísticas e fazer um serviço que utilize o SR para controlar a distribuição de conteúdo de forma mais econômica para a rede P2P e satisfatória para o usuário final.

5.1 Limitações

As limitações desse trabalho são muitas, já que é o primeiro trabalho na área e existem muitas possibilidades relacionadas. As maiores limitações identificadas aqui foram:

- A falta de experimento do modelo com usuários reais não viabiliza o modelo para uso em produção;
- O fator de atenuação β foi criado para possibilitar a exploração de conteúdos ainda não avaliados pelos usuários. Com a variação desse fator ainda não é possível entender o comportamento do modelo, assim como outros fatores, como q e ϵ que estão fixos;
- O banco de dados do MovieLens utilizado nesse trabalho não é apropriado

para comprovar a usabilidade do modelo em um ambiente real.

- O trabalho abrange somente custo de disseminação, sendo assim, outros parâmetros que influenciam QoE precisam ser avaliados para verificar se a diminuição do custo de disseminação pode impactar negativamente em outro parâmetro em um SR e redes reais.

5.2 Trabalhos futuros

Como possíveis trabalhos futuros podem-se apontar trabalhos relativos ao modelo utilizado, à análise dos dados do AG, para criação de um SR real e relativos às novas linhas de pesquisa apresentadas.

No modelo, é possível encontrar uma fórmula que maximize a recompensa com restrição do custo e minimize o custo com restrição da recompensa, é possível utilizar a técnica de multiplicadores de Lagrange ou igualando a derivação das fórmulas de custo e recompensa a zero. Com isso, pode-se obter uma formulação mais simples que combine recompensa e custo, podendo dar origem a novas heurísticas de disseminação de conteúdo.

Ainda no modelo faz-se necessário utilizar um *dataset* maior e melhor para análise do problema estudado. O MovieLens, conforme descrito na seção 3.1, é um grande *dataset*, mas como os usuários não assistiram os filmes exatamente antes de avaliá-los pode não ser muito interessante para o problema estudado. O melhor *dataset* para esse estudo seria o do NetFlix para avaliar parâmetros de precisão e revocação, mas esse não está disponível atualmente. Uma potencial solução é coletar dados de algum roteador central e criar a própria base de dados.

Para melhorar o modelo e encontrar melhores p_{ki}, é possível usar a técnica de *curve fitting* nos resultados obtidos pelo AG para tentar estabelecer heurísticas com base nesses resultados e no que foi observado na seção 4.3.3.

Com essa melhor análise do AG será possível avaliar políticas alternativas para cenários que não foram bem otimizados. A variação do fator de atenuação β já poderia solucionar quase todos os problemas de otimização descritos na seção 4.3.3.

Visando a criação de um SR real, seria necessário mantê-lo utilizando as heurísticas determinadas e adaptável a novas heurísticas. Com isso, o modelo poderia ser colocado em prática e seria adaptável a qualquer nova heurística criada. Também seria possível montar o sistema focado no AG sendo carregadas novas recomendações enquanto o usuário consumisse um conteúdo.

Nesse mesmo SR seria interessante acoplar ao cálculo da recompensa um novo fator para recomendação com base no grau de amizade entre os *peers*, ou algo que possa medir a interação dos usuários durante essas recomendações.

Com essas implementações no SR real, seria possível analisar efeito da recomendação em cada nó do sistema, verificando os reflexos dessa nova forma de entrega de conteúdo com um usuário final para entender o quanto as medições de recompensa ($r_{k,i}$) podem ser melhoradas.

Por último, e que entraria como novas linhas de pesquisa, é a avaliação do cenário de *proxy cache* e ICN com as ferramentas desenvolvidas. Isso seria um novo trabalho utilizando as mesmas ferramentas de avaliação do cenário P2P, mas voltado ao ambiente *proxy cache* e redes de *cache* (como ICN).

REFERÊNCIAS

ADOMAVICIUS, G.; TUZHILIN, A. Towards the Next Generation of Recommender Systems: a survey of the state-of-the-art and possible extensions. **IEEE Transactions on Knowledge and Data Engineering**, [S.l.], v.17, n.6, p.734 – 749, 2005.

AGUIAR RODRIGUES, P. H. de. **Apostila sobre avaliação de desempenho**. `http://www.dcc.ufrj.br/~sadoc/ad20132/apostila.pdf`.

AHLGREN, B.; DANNEWITZ, C. A survey of information-centric networking (draft). **InProceedings**, [S.l.], p.1–26, 2011.

AHLGREN, B. et al. A survey of information-centric networking. **Communications Magazine, IEEE**, [S.l.], v.50, n.7, p.26–36, 2012.

BERKET, K.; ESSIARI, A.; THOMPSON, M. Securing Resources in Collaborative Environments: a peer-to-peer approach. **Proc. of the 17th IASTED International Conference on Parallel and Distributed Computing and Systems**, [S.l.], 2005.

BRADSHAW, M. K. et al. Online scheduling in modular multimedia systems with stream reuse. **Proceedings of the international workshop on Network and operating systems support for digital audio and video (NOSSDAV '05). ACM**, [S.l.], p.195 – 200, 2005.

BURKE, R. Hybrid recommender systems: survey and experiments. **User modeling and user-adapted interaction**, [S.l.], v.12, p.331–370, 2002.

CABANI, A. et al. PHAC: an environment for distributed collaborative applications on p2p networks. In: JANOWSKI, T.; MOHANTY, H. (Ed.). **Distributed**

Computing and Internet Technology. [S.l.]: Springer Berlin Heidelberg, 2007. p.240–247. (Lecture Notes in Computer Science, v.4882).

CAMARILLO, G. **Peer-to-Peer (P2P) Architecture**: definition, taxonomies, examples, and applicability. [S.l.]: IETF, 2009. RFC. (5694).

CERQUEIRA, E. et al. Recent advances and challenges in human-centric multimedia mobile cloud computing. **Computing, Networking and Communications (ICNC), 2014 International Conference on**, [S.l.], p.242–246, Feb 2014.

CHAKOO, N.; GUPTA, R.; HIREMATH, J. Towards Better Content Visibility in Video Recommender Systems. **2008 Japan-China Joint Workshop on Frontier of Computer Science and Technology**, [S.l.], v.201305, p.181–185, Dec. 2008.

CHAN, Y. M. et al. One size doesn't fit all: improving network qos through preference-driven web caching. **Available at SSRN 975735**, [S.l.], 1999.

DACUNTO, L. et al. Peer Selection Strategies for Improved QoS in Heterogeneous BitTorrent-Like VoD Systems. **2010 IEEE International Symposium on Multimedia**, [S.l.], p.89–96, Dec. 2010.

DACUNTO, L.; VINKO, T.; SIPS, H. Bandwidth allocation in BitTorrent-like VoD systems under flashcrowds. **2011 IEEE International Conference on Peer-to-Peer Computing**, [S.l.], p.192–201, Aug. 2011.

DAVIDSON, J. et al. The YouTube Video Recommendation System. **Proceedings of the fourth ACM conference on Recommender systems (RecSys '10). ACM**, [S.l.], p.293–296, 2010.

DRAIDI, F. et al. **P2Prec**: a social-based p2p recommendation system for large-scale data sharing. 2010.

EL-SADDIK, A.; RAHMAN, M.; HOSSAIN, M. Authoring multimedia objects in collaborative ambient intelligent virtual environment. **Haptic Audio Visual Environments and their Applications, 2005. IEEE International Workshop on**, [S.l.], p.6 pp.–, Oct 2005.

ELGAMMAL, A. Human-centered Multimedia: representations and challenges. **Proceedings of the 1st ACM International Workshop on Human-centered Multimedia**, New York, NY, USA, p.11–18, 2006.

GHAZANFAR, M.; PRÜGEL-BENNETT, A. An Improved Switching Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering. **Proceedings of the International MultiConference of Engineers and Computer Scientists**, [S.l.], v.1, 2010.

GHODSI, A.; SHENKER, S.; KOPONEN, T. Information-centric networking: seeing the forest for the trees. **Proceedings of the 10th ACM Workshop on Hot Topics in Networks**, [S.l.], p.1–6, 2011.

GUARNIERI, T. A. et al. Influência do Facebook em Enxames Bittorrent. **XXXII SBRC, Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**, [S.l.], 2014.

GUARNIERI, T.; SILVA, A. P. C. da; VIEIRA, A. B. Impact of Dynamic Social Relationship in a File Sharing System. **Workshop on Dynamic Networks**, [S.l.], 2013.

GUPTA, S.; KAISER, G. P2P video synchronization in a collaborative virtual environment. **Advances in Web-Based Learning–ICWL 2005**, [S.l.], p.1–20, 2005.

HECHT, F. et al. Radiommender: p2p on-line radio with a distributed recommender system. In: **P2P Computing, 2012 IEEE 12th**. [S.l.: s.n.], 2012. p.73–74.

HIRANPONGSIN, S.; BHATTARAKOSOL, P. Integration of recommender system for Web cache management. **Maejo Int. J. Sci. Technol.**, [S.l.], v.7, p.232–247, 2013.

HIRSCH, H.; PEARCE, D. The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. **Speech Recognition: Challenges for the**, [S.l.], 2000.

JAISWAL, N. K. **Priority queues**. [S.l.]: Academic Press New York, 1968. v.50.

KELLERER, W. **Dienstarchitekturen in der Telekommunikation-Evolution, methoden und vergleich**. [S.l.]: Technical Report TUM-LKN-TR-9801, 1998.

KELLY, T. P.; JAMIN, S.; MACKIE-MASON, J. K. Variable QoS from shared Web caches: user-centered design and value-sensitive replacement. **MIT Press**, [S.l.], 2001.

KITCHENHAM, B. Procedures for performing systematic reviews. **Keele, UK, Keele University**, [S.l.], v.33, p.2004, 2004.

KLEINROCK, L. **Queueing systems**: volume 2: computer applications. [S.l.]: John Wiley & Sons New York, 1976. v.82.

KRISHNAN, S. S.; SITARAMAN, R. K. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. In: **Proceedings of the 2012 ACM conference on Internet measurement conference**. [S.l.: s.n.], 2012. p.211–224.

KRISHNAPPA, D. K.; BHAT, D.; ZINK, M. DASHing YouTube: an analysis of using dash in youtube video service. **Local Computer Networks (LCN), 2013 IEEE 38th Conference on**, [S.l.], p.407–415, 2013.

KRISHNAPPA, D. K. et al. Cache-centric Video Recommendation: an approach to improve the efficiency of youtube caches. **Proceedings of the 4th ACM Multimedia Systems Conference**, New York, NY, USA, p.261–270, 2013.

KRISHNAPPA, D. K.; ZINK, M.; GRIWODZ, C. What should you cache?: a global analysis on youtube related video caching. **Proceeding of the 23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video**, [S.l.], p.31–36, 2013.

KUROSE, J. F.; ROSS, K. W. **Computer Networking**: a top-down approach (5th edition). 5.ed. [S.l.]: Addison Wesley, 2010.

MATELL, M. S.; JACOBY, J. Is there an optimal number of alternatives for Likert scale items? I. Reliability and validity. **Educational and psychological measurement**, [S.l.], 1971.

MEI, T. et al. VideoReach: an online video recommendation system. **Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07). ACM**, [S.l.], p.767–768, 2007.

MENASCHE, D. S. et al. Strategic reasoning about bundling in swarming systems. In: **GameNets' 09**. [S.l.: s.n.], 2009. p.611–620.

MENASCHE, D. S.; MASSOULIE, L.; TOWSLEY, D. Reciprocity and Barter in Peer-to-Peer Systems. In: **INFOCOM**. [S.l.: s.n.], 2013.

MENASCHE, D. S.; RODRIGUES, P. A. **Filas com prioridades**. `http://www.dcc.ufrj.br/~sadoc/prioridade/`.

MENDONÇA, G. G.; LEÃO, R. M. M. BTStream — Um ambiente para desenvolvimento e teste de aplicacoes de streaming P2P. **XXX SBRC, Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - Salão de Ferramentas**, [S.l.], Maio 2012.

MOVIELENS. **Movielens - movie recommendations**. [Online; acessado 11 de Março de 2013], `http://movielens.umn.edu/`.

NETFLIX. **Wikipedia Netflix Prize**. [Online; acessado 11 de Março de 2013], `http://en.wikipedia.org/wiki/Netflix_Prize`.

OH, Y. et al. A voice-driven scene-mode recommendation service for portable digital imaging devices. **IEEE Transactions on Consumer Electronics**, [S.l.], v.55, n.4, p.1739–1747, Nov. 2009.

PAGE, L. et al. **The PageRank Citation Ranking**: bringing order to the web. [S.l.]: Stanford InfoLab, 1999. Technical Report, Previous number = SIDL-WP-1999-0120. (1999-66).

PALAU, J. et al. Collaboration analysis in recommender systems using social networks. **Lecture Notes in Computer Science**, [S.l.], v.3191, p.137–151, 2004.

PERINO, D.; VARVELLO, M. A reality check for content centric networking. **Proceeding ICN 11th Proceedings of the ACM SIGCOMM workshop on Information-centric networking**, [S.l.], p.44–49, 2011.

RAO, G. S. Performance Analysis of Cache Memories. **J. ACM**, New York, NY, USA, v.25, n.3, p.378–395, July 1978.

SADDIK, A. et al. PECOLE: p2p multimedia collaborative environment. **Multimedia Tools and Applications**, [S.l.], v.39, n.3, p.353–377, Aug. 2007.

SATYANARAYANA, K.; RAJAGOPALAN, P. S. Fuzzy Classification for Recommender Systems. **International Journal of Soft Computing**, [S.l.], v.5, p.588–591, 2007.

SCHOLLMEIER, R. [16] A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. **Peer-to-Peer Computing, IEEE International Conference on**, [S.l.], p.0101–0101, 2001.

SCHOLLMEIER, R.; HERMANN, F. Topology-analysis of pure peer-to-peer networks. **Kommunikation in Verteilten Systemen (KiVS)**, [S.l.], p.359–370, 2003.

SITARAMAN, R. K. Network performance: does it really matter to users and by how much? In: **COMSNETS, 2013 Fifth International Conference on COMmunication System and NETworkS**. [S.l.: s.n.], 2013. p.1–10.

SPIEGEL, S.; KUNEGIS, J.; LI, F. Hydra: a hybrid recommender system. **Proceeding of the 1st ACM international**, [S.l.], p.75–80, 2009.

STORMER, H.; WERRO, N.; RISCH, D. Recommending Products with a Fuzzy Classification. **CollECTeR Europe 2006**, [S.l.], n.June, 2006.

TRESTIAN, I. et al. Social-Aware Replication in Geo-Diverse Online Systems. **IEEE Transactions on Parallel and Distributed Systems**, Los Alamitos, CA, USA, v.99, n.PrePrints, p.1, 2014.

VIEIRA, D.; DELGADO, C.; MENASCHE, D. Como influenciar seus pares? Sistemas de recomendação de conteúdo para redes P2P. In: **XXXI SBRC, Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - WP2P+**. [S.l.: s.n.], 2013.

VIEIRA, D.; DELGADO, C.; MENASCHE, D. Sistemas de Recomendação de Conteúdo e Seus Impactos no Custo e Desempenho de Redes Par-a-Par. In: **XXXII SBRC, Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**. [S.l.: s.n.], 2014.

WANG, B. et al. Optimal Proxy Cache Allocation for Efficient Streaming Media Distribution. **IEEE Transactions on Multimedia**, [S.l.], v.6, n.2, p.366–374, Apr. 2004.

WU, W.; LUI, J. Exploring the optimal replication strategy in P2P-VoD systems: characterization and evaluation. **Parallel and Distributed Systems, IEEE Transactions on**, [S.l.], v.23, n.8, p.1492–1503, 2012.

ZHANG, H. et al. Can Online Social Friends Help to Improve Data Swarming Performance? **Computer Communications and Networks (ICCCN), 2012 21st International Conference on**, [S.l.], p.1–7, July 2012.

ZHOU, R.; KHEMMARAT, S.; GAO, L. The impact of YouTube recommendation system on video views. **Proceedings of the 10th annual conference on Internet measurement - IMC '10**, New York, New York, USA, p.404, 2010.

ZHOU, Y.; FU, T.; CHIU, D. A unifying model and analysis of p2p vod replication and scheduling. **INFOCOM, 2012 Proceedings IEEE**, [S.l.], 2012.

APÊNDICE A REVISÃO SISTEMÁTICA PARA O TRABALHO

Aplicação da revisão sistemática para verificar se o trabalho realmente é o primeiro da área. Para essa revisão será seguido o seguinte planejamento: descrição do problema, especificar as questões de pesquisa, desenvolver o protocolo de revisão, e avaliar o protocolo de revisão, conforme KITCHENHAM (2004).

A.1 Descrição do problema

Essa dissertação é considerada única publicação no seu ambiente de estudo, exceto os 2 artigos previamente produzidos pelo menos autor. Para verificar a veracidade dessa informação serão buscadas na ferramenta http://google.com as questões de pesquisa indicadas a seguir.

O Google já encontra resultados nas ferramentas mais populares para busca de artigos de computação, incluindo o Google Acadêmico, IEEE Xplore e ACM Digital Library, então não parece necessário buscar em outros lugares.

A.2 Questões de pesquisa

As questões foram divididas tentando sempre buscar a interseção entre sistemas de recomendação e redes P2P, mesmo sabendo que somente essa interseção irá retornar artigos não relacionados como o Radiommender (HECHT et al., 2012).

Devido ao limite de 32 palavras a pesquisa foi dividida em 2 argumentos:

((“sistema de recomendação” OR “recomendação de conteúdo”) AND (custo OR qos OR “qualidade de serviço” OR qoe OR “qualidade de experiência”) AND (p2p OR “par-a-par” OR icn OR “rede orientada a informação” OR “cache” OR cdn OR “rede orientada a conteúdo”)) filetype:pdf

Este argumento de pesquisa retornou 217 resultados, sendo que o Google só informou que somente 39 eram relevantes.

((“recommendation system” OR “content recommendation”) AND (cost OR qoe OR “quality of experience” OR qos OR “quality of service”) AND (“peer-to-peer” OR p2p OR icn OR “information content network” OR “cache” OR cdn OR “content delivery network” OR “content-oriented network”)) filetype:pdf

Este argumento de pesquisa retornou 54.100 resultados, sendo que o Google só informou que somente 288 eram relevantes.

A.3 Protocolo de revisão

A revisão foi feita abrindo cada publicação que se mostrasse interessante pelo título e descrição do Google. Ao abrir foi verificado o item relacionado com as palavras chave e o que desejava encontrar.

A.4 Avaliação do protocolo

Para o primeiro argumento de busca, nenhum dos 217 elementos eram relevantes, sendo que dentro deles estavam os artigos publicados nesse trabalho e os anais dos congressos.

Para o segundo argumento, nenhum dos 54.100 elementos eram relevantes para redes P2P nem para redes orientadas a conteúdo (ROC), na verdade duas páginas depois dos 288 iniciais começaram a aparecer itens bastante estranhos e irrelevantes conforme o Google alertou. Foram encontrados 2 artigos de 2013 (KRISHNAPPA et al., 2013; HIRANPONGSIN; BHATTARAKOSOL, 2013) relativos a utilização de recomendação para melhorar *cache* em aplicações *WEB* como o YouTube, sendo que o primeiro possui 2 citações internacionais do mesmo autor (KRISHNAPPA; ZINK; GRIWODZ, 2013; KRISHNAPPA; BHAT; ZINK, 2013), mas nada com ambientes P2P.

Com isso, é possível avaliar a originalidade e unicidade do tema de estudo para o ambiente acadêmico.

APÊNDICE B SIMULADOR DE *PROXY CACHE* E ICN PARA TRABALHOS FUTUROS

Visando melhor entendimento de SR e servidores *proxy cache*/ICN em trabalhos futuros, foi construído em Python um simulador básico desses servidores. Esse simulador permite a utilização de 4 algoritmos de escalonamento em *caches*: RR (*Random Replacement*), FIFO (*First In, First Out*), LRU (*Least Recently Used*) e LFU (*Least Frequently Used*). O objetivo do simulador é verificar a quantidade de *cache hits* e *cache misses* de um servidor *proxy cache* variando parâmetros como a demanda por cada arquivo. Isso é expansível para ICN, pois podemos considerar uma rede ICN como sendo vários *caches* ligados.

Basicamente, o simulador sorteia um determinado arquivo para ser acessado, se baseando pela frequência de acesso de cada arquivo, que nada mais é do que a probabilidade daquele arquivo ser requisitado. Após sorteio do arquivo, é verificado se o *cache* possui esse arquivo. Caso o arquivo esteja no *cache*, o simulador marca um *hit* e passa para a próxima requisição. Por outro lado, se o arquivo não estiver no *cache*, ele é adicionado e é marcado um *cache miss*. Se o *cache* estiver cheio, um arquivo é removido de acordo com o algoritmo escolhido (LRU, LFU, FIFO ou RR) e isso também é registrado. A Figura B.1 demonstra o cenário descrito utilizando LFU.

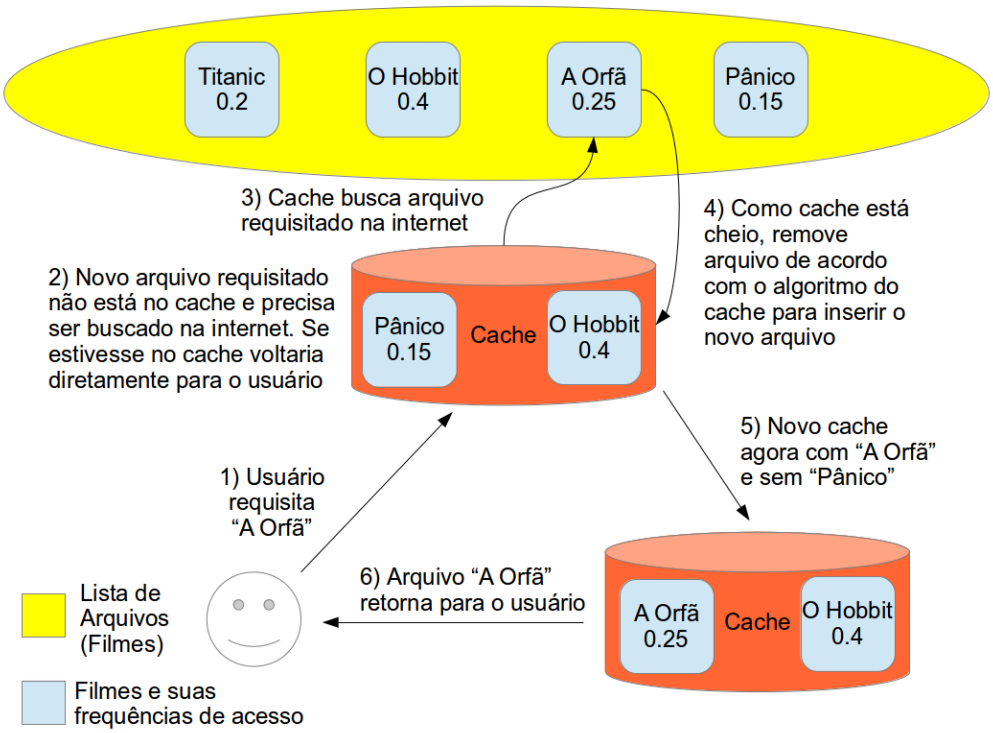


Figura B.1: Ilustração de funcionamento do simulador de *cache* utilizando LFU. Cada arquivo é requisitado com determinada frequência. Caso o arquivo não esteja no *cache*, então ele é encontrado na internet para ser armazenado no *cache* e entregue ao usuário. Se o *cache* estiver cheio, então é removido um arquivo dele conforme a estratégia selecionada para adicionar o novo. O simulador permite analisar a quantidade de *hits* dependendo da estratégia utilizada.

O simulador possui os seguintes parâmetros de entrada:

- **Número de arquivos.** Número total de arquivos na simulação.
- **Número de arquivos no *cache*.** Número máximo de arquivos que o *cache* suporta.
- **Algoritmo do *cache*.** Algoritmo que o *cache* simulado utiliza. Pode ser **LRU**, **LFU**, **FIFO** ou **RR**.

- **Modelo de frequência.** Essa variável preenche a frequência inicial de acesso dos arquivos durante a simulação. É possível escolher entre **aleatório**, **uniforme**, **zipf** ou **particular**; em que nessa última é possível inserir manualmente as frequências para cada arquivo. Caso seja escolhida a distribuição **zipf**, o usuário deve indicar o expoente que será utilizado na distribuição. Esse expoente varia a homogeneidade da distribuição.
- ***Timeslot.*** Quantidade de *timeslots* que a simulação vai ter.
- **Número de requisições.** Número de requisições ao *cache* a cada *timeslot*.

Esse simulador pode ser utilizado para simular uma rede ICN, determinando a quantidade de *caches* que teriam no simulador e o algoritmo desejado para guardar conteúdo.

B.1 Avaliando o impacto da recomendação de conteúdo no *cache hit*

Aqui o objetivo é entender como o sistema de recomendação de conteúdo pode afetar o desempenho do *cache*. Para tal, perturba-se a distribuição de popularidade dos arquivos (i.e., ao aumentar ou diminuir a frequência com que cada arquivo é requisitado), e avaliar o impacto dessa perturbação na frequência de requisições que encontram o conteúdo em *cache* (*cache hit*). Assim sendo, o simulador de *proxy cache* implementado captura o efeito de sistemas de recomendação de conteúdo. Para caracterizar o quanto que os usuários irão deixar-se influenciar pelo SR, foi incluído um parâmetro adicional em nosso simulador:

- **Susceptibilidade da demanda.** Essa é a quantidade de requisições por cada arquivo que é sensível ao sistema de recomendação. Permite a modificação da

frequência de requisições dos arquivos, simulando a possibilidade de um usuário aceitar ou não determinada recomendação.

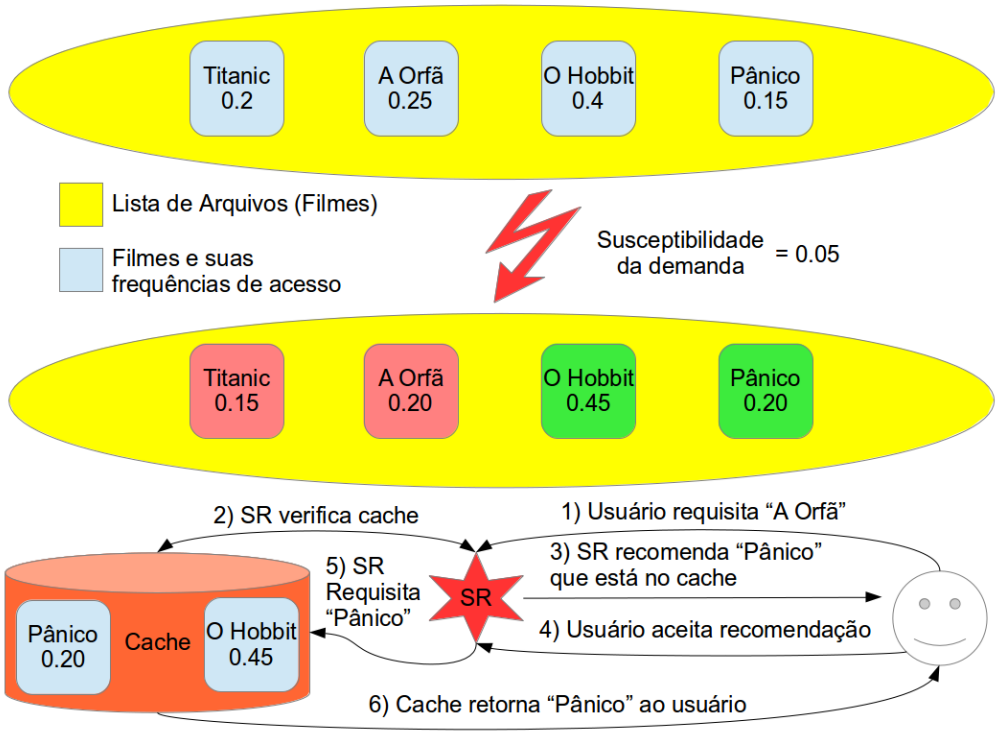


Figura B.2: Funcionamento do simulador de *cache* implementado com modificação nas frequências de requisições dos usuários. Os arquivos possuem uma frequência de requisições e uma pequena susceptibilidade da demanda que modifica essa frequência. Com essa modificação na frequência, é possível utilizar um Sistema de Recomendação (SR) para simular o efeito de uma recomendação de conteúdo, aumentando a chance de *cache hit*.

Com essa **susceptibilidade da demanda** será possível entender como que as preferências dos usuários de uma rede podem interferir no desempenho do cache. Uma requisição ao item j que seja atendida com item i pode acarretar ganhos no tempo de resposta e maior quantidade de *hits* no *cache*. Uma vez definida a **susceptibilidade da demanda**, é necessário definir como que o sistema de recomendação de conteúdo irá efetivamente afetar essa demanda, mas isso ainda está em aberto para trabalhos futuros.

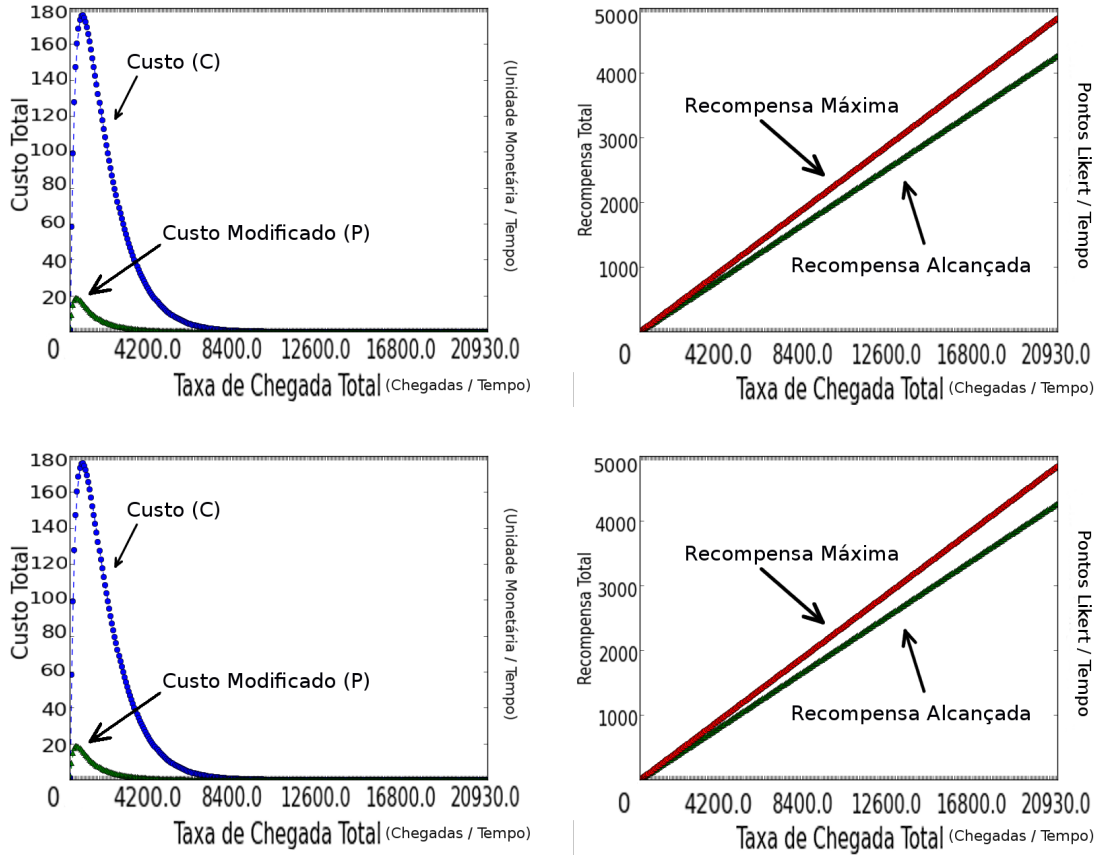
A Figura B.2 ilustra o funcionamento desse simulador de *proxy cache*. Nessa figura admite-se que o SR possui conhecimento sobre o conteúdo contido no *cache*, mas isso também pode não ser verdade e uma alternativa é otimizar a distribuição de conteúdo utilizando o caso médio.

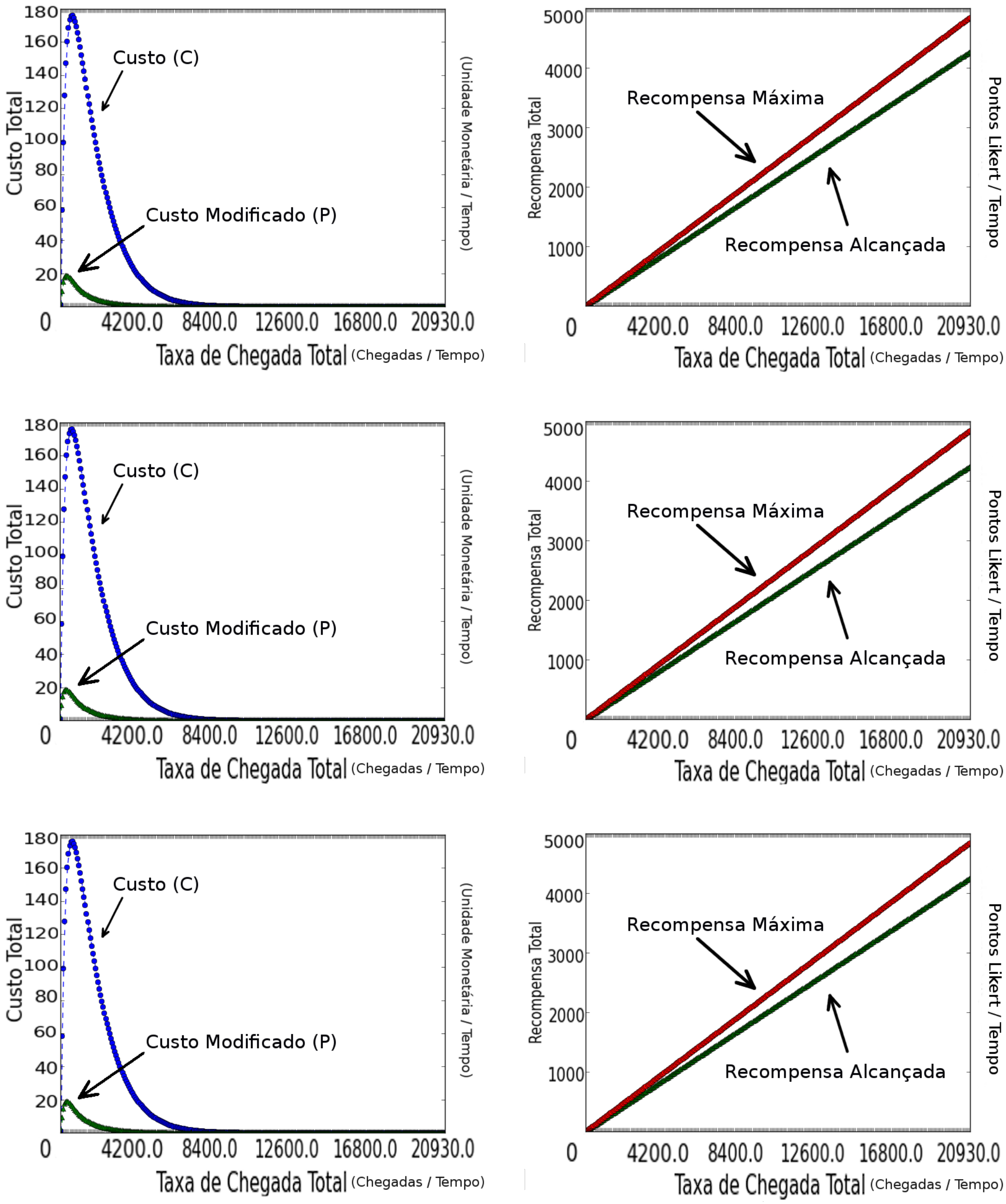
Como resultado do simulador é possível analisar a quantidade de *hits* e *misses* no *cache* por arquivo, e com isso será possível ter conhecimento sobre como uma pequena influência nas probabilidades de acesso dos arquivos em um *proxy cache*, devido a um sistema de recomendação, pode modificar seu desempenho em termos de *hits* e *misses*.[1]

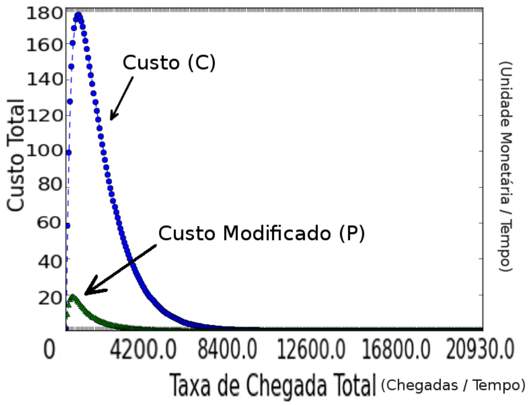
[1]Mais detalhes e o código fonte do simulador *proxy cache* estão disponíveis em: `https://github.com/dmvieira/SimuCache`

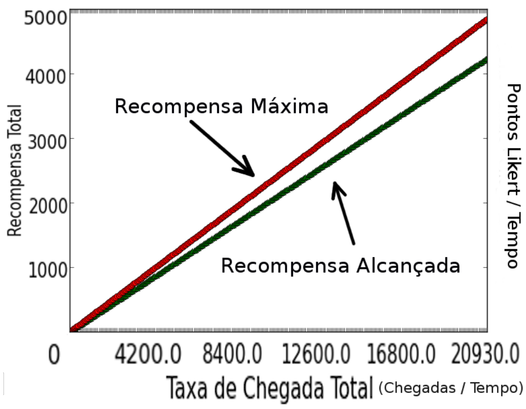
APÊNDICE C DADOS DOS RESULTADOS COM ALGORITMO GENÉTICO

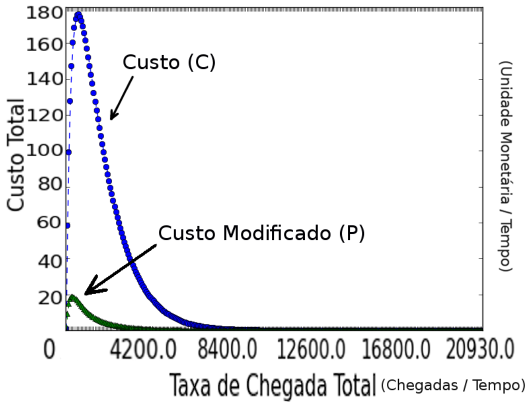
Nesse apêndice estão os dados que levaram as análises dos resultados do AG. A Figura C.1 mostra o quão próximos são os resultados obtidos independentemente da proporção de $\overline{C}$ e R na função de *Fitness* (3.18), conforme seção 3.4.2. A seguir, está a Tabela C.1 com os 87 genes utilizados no AG e os respectivos $p_{k,i}$ determinados pelo AG.

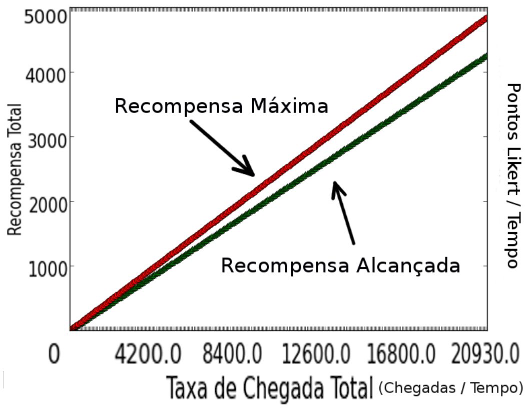


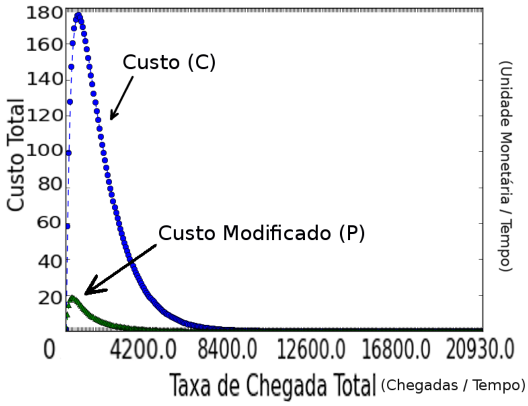

Custo (C)
Custo Modificado (P)
Custo Total
(Unidade Monetária / Tempo)
Taxa de Chegada Total (Chegadas / Tempo)
0
4200.0
8400.0
12600.0
16800.0
20930.0
Recompensa Máxima
Recompensa Alcançada
Recompensa Total
Pontos Likert / Tempo

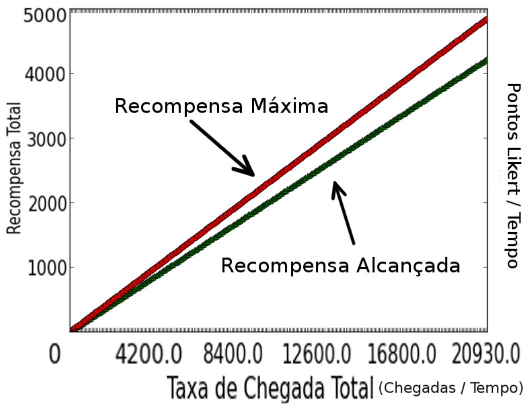

Custo Total
(Unidade Monetária / Tempo)
Custo (C)
Custo Modificado (P)
180
160
140
120
100
80
60
40
20
0
4200.0
8400.0
12600.0
16800.0
20930.0
Taxa de Chegada Total (Chegadas / Tempo)


Recompensa Total
Pontos Likert / Tempo
Recompensa Máxima
Recompensa Alcançada
5000
4000
3000
2000
1000
0
4200.0
8400.0
12600.0
16800.0
20930.0
Taxa de Chegada Total (Chegadas / Tempo)


Custo Total
(Unidade Monetária / Tempo)
Custo (C)
Custo Modificado (P)
180
160
140
120
100
80
60
40
20
0
4200.0
8400.0
12600.0
16800.0
20930.0
Taxa de Chegada Total (Chegadas / Tempo)


Recompensa Total
Pontos Likert / Tempo
Recompensa Máxima
Recompensa Alcançada
5000
4000
3000
2000
1000
0
4200.0
8400.0
12600.0
16800.0
20930.0
Taxa de Chegada Total (Chegadas / Tempo)


Custo Total
(Unidade Monetária / Tempo)
Custo (C)
Custo Modificado (P)
180
160
140
120
100
80
60
40
20
0
4200.0
8400.0
12600.0
16800.0
20930.0
Taxa de Chegada Total (Chegadas / Tempo)


Recompensa Total
Pontos Likert / Tempo
Recompensa Máxima
Recompensa Alcançada
5000
4000
3000
2000
1000
0
4200.0
8400.0
12600.0
16800.0
20930.0
Taxa de Chegada Total (Chegadas / Tempo)

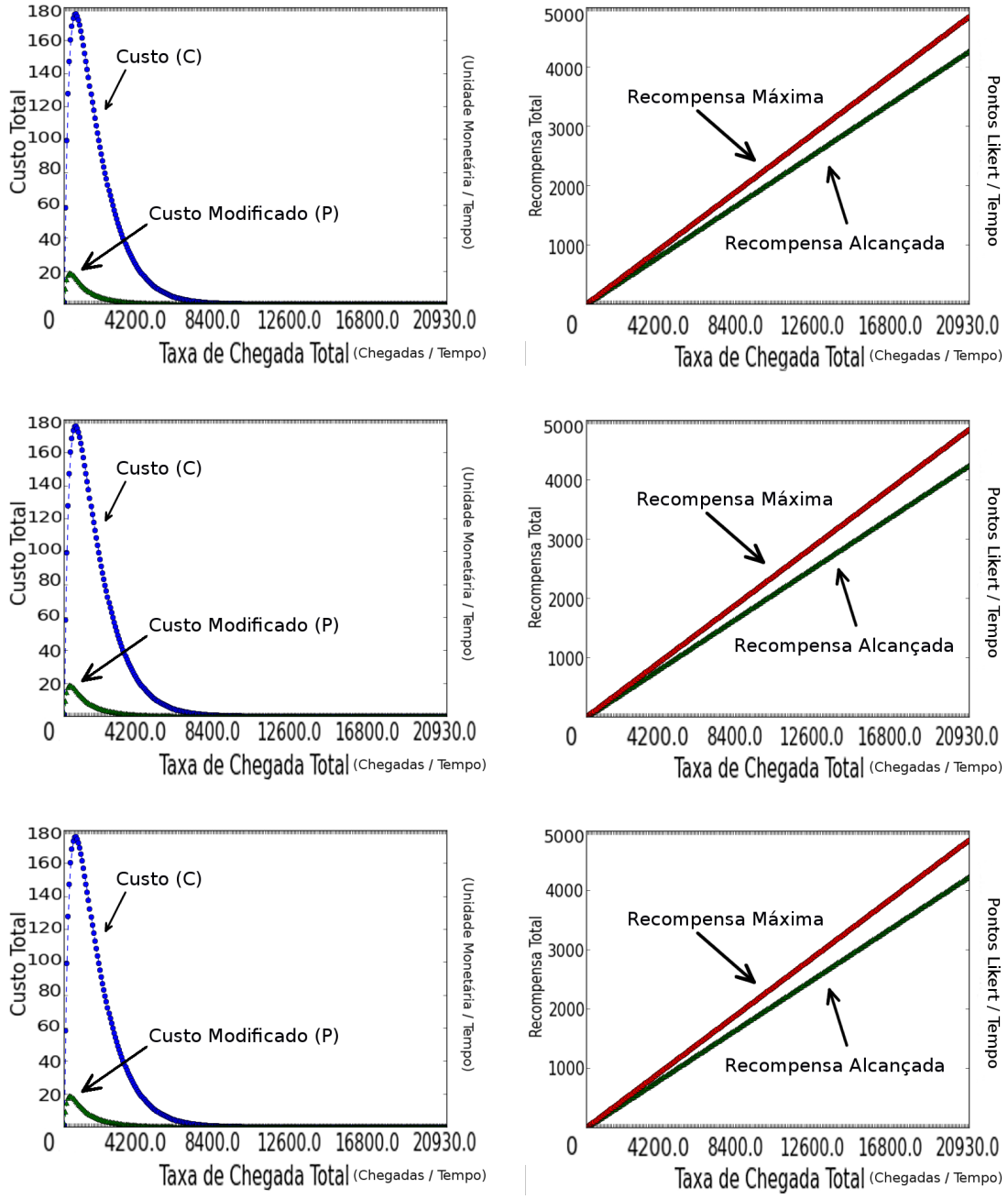


Figura C.1: Tradeoffs envolvidos no sistema de recomendação com AG (custos e recompensas) para proporções de custo e recompensa. Comparado com os resultados da Figura 4.4 não houve grandes mudanças ao variar os valores de $\overline{C}$ e R na função de *Fitness* (3.18), conforme seção 3.4.2. Os resultados estão em pares (custo e recompensa) ordenados conforme a sequência: $\{(R\times 1, \overline{C}\times 0), (R\times 0.9, \overline{C}\times 0.1) \ldots (R\times 0, \overline{C}\times 1)\}$

Nessa tabela, são comparados lado a lado cada resultado obtido em diferentes rodadas do AG. Conforme verificado nos resultados das Figuras C.1 originados a partir dessas rodadas, todos os resultados são bem parecidos com custos baixos e bem próximos da maior recompensa possível. Na tabela aparecem também resultados bem parecidos de $p_{k,i}$, mostrando que o AG foi bem sucedido na otimização para diversas combinações de variáveis.

A partir da Tabela C.1 foi identificado que apesar da pequena diferença entre as otimizações a combinação de parâmetros $\overline{\boldsymbol{C}}\mathbf{0.8}\ \boldsymbol{R}\mathbf{0.2}$ foi mais bem sucedida com o AG.

λ_k	k	i	$r_{k,i}$	inicial	$\overline{C}$0 R1	$\overline{C}$0.1 R0.9	$\overline{C}$0.2 R0.8	$\overline{C}$0.3 R0.7	$\overline{C}$0.4 R0.6	$\overline{C}$0.5 R0.5	$\overline{C}$0.6 R0.4	$\overline{C}$0.7 R0.3	$\overline{C}$0.8 R0.2	$\overline{C}$0.9 R0.1	$\overline{C}$1 R0
2	8	197	1	1	1	1	1	1	1	1	1	1	**1**	1	1
4	20	265	0.3333333333	0.8	0.8	0.8	0.8	0.86	0.91	0.87	0.94	0.79	**0.93**	0.93	0.93
3	54	66	1	1	1	1	1	1	1	1	1	1	**1**	1	1
7	56	198	0.5	0.18	0.18	0.18	0.18	0.12	0.67	0.38	0.67	0.04	**0.18**	0.18	0.35
1	66	77	1	1	1	1	1	1	1	1	1	1	**1**	1	1
3	86	443	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	88	1042	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	91	755	0.4285714286	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.69	0.06	**0.06**	0.06	0.37
7	98	198	0.5	0.82	0.82	0.82	0.82	0.88	0.33	0.62	0.33	0.96	**0.82**	0.82	0.65
6	117	202	0.3333333333	0.95	0.95	0.95	0.95	0.95	0.73	0.95	1	0.95	**0.95**	0.95	0.97
5	121	1060	0.5	0.87	0.87	0.96	0.96	0.87	0.96	0.87	0.96	0.87	**0.87**	1	0.81
5	121	927	0.5	1	1	1	0.98	0.81	0.98	0.81	0.98	0.81	**0.81**	0.88	0.73
1	129	558	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	129	447	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	129	164	1	1	1	1	1	1	1	1	1	1	**1**	1	1
2	133	176	1	1	1	1	1	1	1	1	1	1	**1**	1	1
5	135	234	1	1	1	1	1	1	1	1	1	1	**1**	1	1
5	135	460	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	156	222	0.3333333333	0.23	0.23	0.4	0.01	0.08	0.43	0.33	0.13	0.15	**0.13**	0.16	0.13
2	161	88	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	162	1016	0.3333333333	0.58	0.58	0.44	0.18	0.37	0.69	0.92	0.79	0.24	**0.79**	0.06	0.79
2	164	440	0.3333333333	0.27	0.27	0.08	0.92	0.09	0.43	0.08	0.44	0.11	**0.44**	0.47	0.44
5	172	742	1	1	1	1	1	1	1	1	1	1	**1**	1	1
2	173	133	1	1	1	1	1	1	1	1	1	1	**1**	1	1
2	173	366	1	1	1	1	1	1	1	1	1	1	**1**	1	1
3	187	1184	1	1	1	1	1	1	1	1	1	1	**1**	1	1
3	187	487	1	1	1	1	1	1	1	1	1	1	**1**	1	1
3	187	649	1	1	1	1	1	1	1	1	1	1	**1**	1	1
4	194	222	0.3333333333	0	0	0	0.32	0.39	0.3	0.17	0.17	0	**0.17**	0	0.17
2	197	944	1	1	1	1	1	1	1	1	1	1	**1**	1	1
2	198	836	1	1	1	1	1	1	1	1	1	1	**1**	1	1
4	202	160	1	1	1	1	1	1	1	1	1	1	**1**	1	1
3	217	182	1	1	1	1	1	1	1	1	1	1	**1**	1	1
3	217	955	1	1	1	1	1	1	1	1	1	1	**1**	1	1
5	222	873	1	1	1	1	1	1	1	1	1	1	**1**	1	1
4	228	222	0.3333333333	0.77	0.77	0.6	0.67	0.53	0.27	0.5	0.7	0.85	**0.7**	0.84	0.7
5	234	202	0.3333333333	0	0	0	0	0	0	0	0	0	**0**	0	0
1	240	217	0.3333333333	0.48	0.48	0.37	0.37	0.94	0.37	0.22	0.22	0.93	**0.48**	0.37	0.94
1	257	1016	0.3333333333	0.35	0.35	0	0.5	0	0.25	0	0	0.2	**0**	0.38	0
6	258	298	0.5	0.99	0.99	0.92	0.92	0.92	0.92	0.87	0.87	0.99	**0.99**	0.99	0.92
6	258	498	0.3333333333	0.88	0.88	0.88	0.88	0.88	0.88	0.81	0.81	0.6	**0.98**	0.9	0.88
5	265	1028	0.3333333333	0.92	0.92	0.89	0.86	0.86	0.86	0.84	0.84	0.92	**0.87**	0.79	0.86
5	265	150	1	1	1	1	1	1	1	1	1	1	**1**	1	1
5	274	1028	0.3333333333	0	0	0	0	0	0	0	0	0	**0**	0	0
2	298	237	1	1	1	1	1	1	1	1	1	1	**1**	1	1
6	304	172	0.3333333333	0.78	0.78	0.11	0.9	0.78	0.78	0.42	0.11	0.97	**0.42**	0.69	0.25
1	306	1288	1	1	1	1	1	1	1	1	1	1	**1**	1	1
4	317	486	1	1	1	1	1	1	1	1	1	1	**1**	1	1
2	325	306	1	1	1	1	1	1	1	1	1	1	**1**	1	1
7	328	172	0.3333333333	0.08	0.08	0.73	0	0.08	0.08	0.53	0.73	0	**0.53**	0	0.73
3	346	688	0.5	0.93	0.93	0.87	0.93	0.87	0.89	0.82	0.87	0.93	**0.82**	0.74	0.74
2	347	688	0.5	0.07	0.07	0.13	0.07	0.13	0.11	0.18	0.13	0.07	**0.18**	0.26	0.26
1	366	755	0.5714285714	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.31	0.94	**0.94**	0.94	0.63
4	382	497	1	1	1	1	1	1	1	1	1	1	**1**	1	1
4	382	1198	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	431	217	0.3333333333	0.08	0.08	0.34	0.34	0	0.34	0	0	0	**0.08**	0.34	0
3	443	208	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	447	440	0.3333333333	0.39	0.39	0	0	0.12	0.15	0.36	0	0.16	**0**	0.09	0
1	460	202	0.3333333333	0.05	0.05	0.05	0.05	0.05	0.27	0.05	0	0.05	**0.05**	0.05	0.03
3	486	281	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	487	632	0.5	0.36	0.36	0.38	0.36	0.12	0.52	0.28	0.38	0.93	**0.71**	0.39	0.39
1	492	382	0.4444444444	0.37	0.37	0.16	0.37	0.16	0.09	0.29	0.24	0.84	**0.27**	0.68	0.68
3	498	58	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	505	265	0.3333333333	0	0	0	0	0	0	0	0	0.14	**0.01**	0.01	0.01
1	506	382	0.5555555556	0.63	0.63	0.84	0.63	0.84	0.91	0.71	0.76	0.16	**0.73**	0.32	0.32
1	511	498	0.3333333333	0	0	0	0	0	0	0	0	0.28	**0**	0	0
3	546	1028	0.3333333333	0.08	0.08	0.11	0.14	0.14	0.14	0.16	0.16	0.08	**0.13**	0.21	0.14
2	558	440	0.3333333333	0.34	0.34	0.92	0.08	0.79	0.42	0.56	0.56	0.73	**0.56**	0.44	0.56
2	649	1197	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	678	1016	0.3333333333	0.07	0.07	0.56	0.32	0.63	0.06	0.08	0.21	0.56	**0.21**	0.56	0.21
1	680	172	0.3333333333	0.14	0.14	0.16	0.1	0.14	0.14	0.05	0.16	0.03	**0.05**	0.31	0.02
2	688	96	0.5	0.1	0.1	0.05	0.09	0.04	0.04	0.04	0.05	0.04	**0.01**	0.04	0.04
2	688	498	0.3333333333	0.12	0.12	0.12	0.12	0.12	0.12	0.19	0.19	0.12	**0.02**	0.1	0.12
2	746	217	0.3333333333	0.44	0.44	0.29	0.29	0.06	0.29	0.78	0.78	0.07	**0.44**	0.29	0.06
2	755	1047	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	870	1060	0.5	0.13	0.13	0.04	0.04	0.13	0.04	0.13	0.04	0.13	**0.13**	0	0.19
1	870	927	0.5	0	0	0	0.02	0.19	0.02	0.19	0.02	0.19	**0.19**	0.12	0.27
1	877	298	0.5	0.01	0.01	0.08	0.08	0.08	0.08	0.13	0.13	0.01	**0.01**	0.01	0.08
1	927	741	0.5	0.09	0.09	0.09	0.09	0.09	0.13	0.19	0.09	0.97	**0.22**	0.96	0.83
2	955	583	1	1	1	1	1	1	1	1	1	1	**1**	1	1
3	1016	21	1	1	1	1	1	1	1	1	1	1	**1**	1	1
5	1028	96	0.5	0.9	0.9	0.95	0.91	0.96	0.96	0.96	0.95	0.96	**0.99**	0.96	0.96
2	1060	741	0.5	0.91	0.91	0.91	0.91	0.91	0.87	0.81	0.91	0.03	**0.78**	0.04	0.17
2	1067	265	0.3333333333	0.2	0.2	0.2	0.2	0.14	0.09	0.13	0.06	0.07	**0.06**	0.06	0.06
2	1098	1099	1	1	1	1	1	1	1	1	1	1	**1**	1	1
2	1099	485	1	1	1	1	1	1	1	1	1	1	**1**	1	1
1	1184	632	0.5	0.64	0.64	0.62	0.64	0.88	0.48	0.72	0.62	0.07	**0.29**	0.61	0.61

Tabela C.1: Tabela com valores obtidos de $p_{k,i}$ nas rodadas de AG para a primeira rodada sem restrições (inicial) e para as rodadas com combinações de $\overline{C}$ e R da função de *Fitness* (3.18), conforme seção 3.4.2. Os valores foram obtidos a partir dos λ_k e $r_{k,i}$ dos conteúdos. A coluna em negrito (**$\overline{C}$0.8 R0.2**) teve os melhores resultados.