

CHARLES CHRISTIAN MIERS

**UMA ARQUITETURA USANDO TRACKERS
HIERÁRQUICOS PARA LOCALIDADE EM REDES
P2P GERENCIADAS**

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção do
Título de Doutor em Ciências.

São Paulo
2012

CHARLES CHRISTIAN MIERS

**UMA ARQUITETURA USANDO TRACKERS
HIERÁRQUICOS PARA LOCALIDADE EM REDES
P2P GERENCIADAS**

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Doutor em Ciências.

Área de Concentração:
Engenharia Elétrica
Sistemas Digitais

Orientadora:
Prof^a. Dr^a. Tereza Cristina Melo de Brito
Carvalho

São Paulo
2012

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 19 de dezembro de 2012.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Miers, Charles Christian

Uma arquitetura usando trackers hierárquicos para localidade em redes P2P gerenciadas / C.C. Miers. -- ed.rev. -- São Paulo, 2012.

185 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1. Redes de computadores 2. Sistemas multimídia 3. Sistemas distribuídos I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II. t.

“Desde que me cansei de procurar,
aprendi a encontrar;
Desde que o vento me opõe resistência,
velejo com todos os ventos.”

Friedrich Nietzsche

AGRADECIMENTOS

À minha orientadora, Professora Tereza Cristina Melo de Brito Carvalho, pela orientação e também por ensinar uma forma diferente de ver as pessoas e o mundo. A admiração e o respeito que sinto por minha orientadora a coloca em uma posição de grande destaque na minha vida.

À minha esposa Jani Floriano por compreender todo o esforço e sacrifício necessário por trás dessa etapa da minha vida. O apoio e a compreensão foram determinantes para a realização desta tese.

À minha família (Auria, Dieter e James) que sempre reconheceu o valor dos estudos e viu na minha capacitação uma forma de contribuir para com o progresso da minha cidade e da humanidade.

Aos meus avós Augusto e Edith Braatz (*in memorian*) por sempre me incentivarem nos estudos.

Aos meus colegas de pós-graduação e do LARC (Laboratório de Arquitetura e Redes de Computadores), em especial a: Simplicio, Diego, Vlad, Akio, Pedro, Nelson, Fernando, Rony, Cristina, Marcelo, Torrez e Carlos. A discussão científica e amizade ajudaram a forjar o meu trabalho, engrandecendo esta minha experiência de vida.

À Ericsson Telecomunicações do Brasil, pelo suporte financeiro (através da FDTE) e experiência em pesquisa orientada ao mercado. Meus agradecimentos em especial ao Victor Souza e Mats Näslund que muito contribuíram com sugestões no desenvolvimento do meu trabalho e nos projetos que estive envolvido. Também a minha gratidão à Maria Valéria Marquezini, Makan Pourzandi, Jan-Erik Mangs, Per Karlsson e Ayodele Damola.

À FDTE (Fundação para o Desenvolvimento Tecnológico da Engenharia) pelo suporte financeiro e apoio organizacional em todos os projetos que estive envolvido. Agradeço em especial à Edilaine, que sempre buscou auxiliar de todas as maneiras possíveis a viabilização dos recursos envolvidos nos projetos que atuei.

Agradeço também a UDESC (Universidade do Estado de Santa Catarina), mais em específico aos meus colegas do CCT/DCC (Centro de Ciências Tecnológicas / Departamento de Ciência da Computação), por todo o auxílio prestado e o incentivo para realizar esta capacitação.

RESUMO

As redes *Peer-to-Peer* (P2P) tornaram-se nos últimos anos um método atrativo para a distribuição de conteúdo multimídia através da Internet. Muitos fatores contribuíram para esse sucesso, mas os custos baixos de distribuição e a escalabilidade inerente das redes P2P, na qual os consumidores de conteúdo também são fontes potenciais, estão entre os mais proeminentes. Entretanto, a efetividade e desempenho de várias redes P2P populares (como as que usam o protocolo BitTorrent) é consideravelmente dependente de quanto bem o *tracker* (que é o elemento responsável pela identificação e gerência dos participantes de uma rede P2P do tipo BitTorrent) seleciona os *peers* que irão fornecer o conteúdo. Os *peers* escolhidos pelo *tracker* irão afetar diretamente a percepção do usuário sobre o desempenho do serviço e o uso dos recursos de rede. Além disso, as redes P2P usualmente não tem percepção de localidade, resultando em uma utilização não otimizada dos recursos de rede. A fim de abordar estas questões, nesta tese é apresentada uma proposta inédita de uma arquitetura de *trackers* hierárquicos para localidade orientada a redes P2P gerenciadas e baseadas no protocolo BitTorrent. Na tese é identificado, através de experimentação, que o uso da arquitetura proposta conduz a uma melhora significativa no desempenho da rede sem comprometer a experiência do usuário. Dentre as melhorias, a principal é o controle do tráfego de dados trafegado entre os *peers* através da escolha dos *peers* feita pelo *tracker* com base em informações da rede e regras de negócio. Essa melhoria permite que a rede possa ser gerenciada de maneira pró-ativa e o dinamismo de adaptação da rede às condições adversas possa ser obtido por meio de políticas de configuração que são acionadas por gatilhos pré-determinados. Nesta tese são usados gatilhos baseados em tempo (data/horário) para exemplificar a abordagem de mudança de políticas programável.

Palavras-chave: Localidade. BitTorrent. *Trackers* Hierárquicos.

ABSTRACT

Peer-to-Peer (P2P) networks have become an attractive method for distributing multimedia content over the Internet in the last years. Several factors contributed to this success, but the low distribution costs and the inherent scalability of P2P networks, in which content consumers are also potential sources, are among of the most prominent. However, the effectiveness and performance of several popular P2P networks (such as those that use the BitTorrent protocol) is considerably dependent on how well the tracker (which is the element responsible for the identification and management of participants in BitTorrent P2P networks) selects peers that will provide the content. The peers chosen by the tracker will directly affect the user's perception about the service performance and proper deployment of the network resources. In addition, P2P networks usually have no sense of locality, resulting in a non-optimal utilization of network resources. In order to address these issues, this thesis presents a novel hierarchical tracker architecture using P2P locality for managed networks based on a modified version of the BitTorrent protocol. This thesis shows, through experimentation, that the adoption of the proposed architecture leads to significant network efficiency improvements without compromising end-user experience. Among the improvements, the principal is the data flow control between the peers through the choice of peers given by tracker based on network information and business rules. This enhancement allows to manage the network proactively and to perform dynamic adaptation of the network due to adverse conditions. The network control can be achieved by setting policies that are driven by predetermined triggers. This thesis uses time-based triggers (date / time) to exemplify the approach of programmable policy change.

Keywords: Locality. BitTorrent. Hierarchical Trackers.

SUMÁRIO

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

1	Introdução	18
1.1	Motivação	19
1.2	Descrição do Problema	21
1.3	Objetivos	22
1.4	Método	23
1.5	Organização do trabalho	25
2	Conceitos Básicos	27
2.1	Redes P2P	27
2.2	Histórico das redes P2P	32
2.3	Sistemas P2P BitTorrent	36
2.3.1	Componentes de um Sistema BitTorrent	37
2.3.2	Etapas de Obtenção de Conteúdo	39
2.3.3	Critérios de Seleção de Pedaços	40
2.3.4	Critérios de Seleção de um <i>Peer</i>	41

2.3.5	Problemas nos Sistemas BitTorrent	43
2.4	Considerações do Capítulo	45
3	Localidade em Redes P2P	47
3.1	Breve histórico da localidade em redes P2P	48
3.2	Métricas de localidade em redes P2P	50
3.3	Classificação de localidade em redes P2P	53
3.4	Principais algoritmos de localidade para redes P2P	56
3.5	Considerações do Capítulo	59
4	Proposta do Sistema P2PM	61
4.1	Especificação de requisitos	63
4.1.1	Requisitos funcionais	64
4.1.2	Requisitos não funcionais	65
4.2	Descrição da arquitetura proposta	65
4.2.1	Domínio <i>Tracker</i>	67
4.2.2	Agregação dos dados	69
4.3	Implementação da solução	70
4.3.1	Modelo do <i>Tracker</i>	71
4.3.2	Computação do peso das arestas do grafo	75
4.3.2.1	Algoritmo de menor caminho	77
4.3.3	Hierarquia de <i>trackers</i> e agregação dos dados	79
4.3.4	Manutenção dos grafos	84

4.4	Caso de uso	89
4.4.1	Obtenção de conteúdo	90
4.5	Limitações da solução proposta	91
4.6	Trabalhos relacionados	93
4.7	Considerações do Capítulo	96
5	Ambiente de Experimentação	98
5.1	Sistema de IPTV baseado em DHT	98
5.2	Sistema IPTV com serviço de localidade	102
5.2.1	Implementação	104
5.3	Configuração do Ambiente de Teste	106
5.4	Limitações da implementação	111
5.5	Considerações do Capítulo	112
6	Experimentos & Análise dos Resultados	114
6.1	Testes	117
6.1.1	Teste CS 0 - Cliente-servidor	120
6.1.2	Teste P2P 0 - BitTorrent padrão	122
6.1.3	Teste P2P 1 - Conteúdo disponível no domínio da solicitação .	125
6.1.4	Teste P2P 2 - Conteúdo indisponível no domínio da solicitação	127
6.1.5	Teste P2P 3 - Mudança nas condições da rede	130
6.1.6	Teste P2P 4 - Substituição do perfil do grafo no <i>Tracker1</i> . . .	133
6.2	Análise dos resultados	136

6.2.1	Análise do tempo de inicialização, tempo médio para a obtenção dos blocos e interrupção da apresentação	137
6.2.2	Controle do uso dos recursos de rede	139
6.2.3	Análise de requisitos <i>versus</i> resultados	141
6.3	Considerações do Capítulo	143
7	Considerações Finais	144
7.1	Contribuições e Inovações	147
7.2	Trabalhos Futuros	149
Referências		151
Apêndice A - Publicações		161
Apêndice B - Arquivo XML dos perfis dos grafos		165
B.1	<i>Tracker1</i>	166
B.2	<i>Tracker1</i> com prioridades alteradas	170
B.3	<i>Tracker2</i>	174
B.4	<i>Root Tracker</i>	177
Apêndice C - Interface de monitoração em tempo real		180

LISTA DE FIGURAS

1	Modelo centralizado cliente-servidor e modelo distribuído P2P	28
2	Estatística do tipo de tráfego nos principais <i>backbones</i> de rede dos EUA. Fonte: (IPOQUE, 2007)	34
3	Composição do tráfego agregado na Internet no período entre os anos de 2008 a 2010.Fonte:(CAIDA, 2010).	35
4	Composição do tráfego agregado na Internet na América do Norte entre os anos de 2009 a 2012.Fonte:(SANDVINE, 2012).	36
5	Funcionamento de um sistema BitTorrent básico	39
6	Exemplo de arquiteturas que podem ser empregadas para obter determinadas características de um sistema BitTorrent	44
7	Classificação de localidade para redes P2P	53
8	Abordagem para otimizar redes usando engenharia de tráfego de dados. Adaptado de: (REXFORD, 2006)	67
9	Hierarquia de <i>trackers</i>	68
10	Grafo do Domínio <i>Tracker2</i> , enlaces de rede assimétricos	72
11	Diagrama de fluxo do tratamento das requisições do <i>peers</i> pelo <i>tracker</i> empregando localidade	73
12	Domínios dos <i>trackers</i> e <i>Root Tracker</i>	81
13	Máquina de estados da manutenção do grafo nos <i>trackers</i>	85
14	Máquina de estados da manutenção do grafo no <i>Root Tracker</i>	87

15	Verificação dos elementos listados no perfil do grafo no <i>Root Tracker</i>	88
16	Visão geral do sistema de IPTV baseado em <i>Distributed Hash Tables</i> (DHT). Adaptado de: (GALLO et al., 2009)	99
17	Transformação do conteúdo em para uso no BitTorrent	101
18	Disposição dos elementos do sistema IPTV e topologia da rede	107
19	Interface de monitoração do sistema em tempo real	110
20	Interface da monitoração das arestas do sistema	111
21	Identificação dos enlaces do <i>testbed</i>	114
22	Teste CS 0: Disposição dos elementos do sistema e topologia da rede .	121
23	Teste CS 0 - Quantidade de dados trafegados nos enlaces	121
24	Vazão durante a obtenção do conteúdo no Teste CS 0.	122
25	Teste P2P 0: Disposição dos elementos do sistema IPTV e topologia da rede	123
26	Teste P2P 0 - Quantidade de dados trafegados nos enlaces	123
27	Teste P2P 0 - Vazão por <i>peers</i>	124
28	Teste P2P 1 - Disposição dos elementos do sistema IPTV e topologia da rede	125
29	Teste P2P 1 - Quantidade de dados trafegados nos enlaces	126
30	Teste P2P 1 - Vazão por <i>peers</i>	126
31	Teste P2P 1 - Tráfego de controle entre <i>Peer1</i> e <i>Tracker1</i>	127
32	Teste P2P 2 - Disposição dos elementos do sistema IPTV e topologia da rede	128
33	Teste P2P 2 - Quantidade de dados trafegados nos enlaces	128

34	Teste P2P 2 - Vazão por <i>peers</i>	129
35	Teste P2P 3 - Disposição dos elementos do sistema IPTV e topologia da rede	130
36	Teste P2P 3 - Quantidade de dados trafegados nos enlaces	131
37	Teste P2P 3 - Vazão por <i>peers</i>	132
38	Teste P2P 3 - Alternância entre uso de <i>peers</i> em função das condições da rede	132
39	Teste P2P 4 - Disposição dos elementos do sistema IPTV e topologia da rede	133
40	Teste P2P 4 - Substituição do perfil do grafo no <i>Tracker1</i>	134
41	Teste P2P 4 - Vazão por <i>peers</i>	135
42	Teste P2P 4 - Alternância entre uso de <i>peers</i> em função das mudança do perfil do grafo do <i>Tracker1</i>	135
43	Comparação Teste CS 0, Teste P2P 0 e Teste P2P 1: Dados trafegados por enlace	140
44	Interface de monitoração do sistema em tempo real	181
45	Interface de monitoração: Nome do Nó	181
46	Interface de monitoração: ID do Nó	182
47	Interface de monitoração: Nome do enlace	182
48	Interface de monitoração: Banda em uso	183
49	Interface de monitoração: % do enlace em uso	183
50	Interface de monitoração: peso atual da aresta	184
51	Interface de monitoração: listas de <i>peers</i> retornados	184

LISTA DE TABELAS

1	Classificação das Redes P2P. Adaptado de (BOUTABA, 2008)	32
2	Métricas de localidade.	52
3	Classificação dos algoritmos de localidade.	58
4	Exemplo de Tabela de Conteúdo dos <i>trackers</i> para o domínio do <i>Tracker1</i>	82
5	Exemplo de Tabela de Conteúdo dos <i>trackers</i> para o domínio do <i>Tracker2</i>	82
6	Exemplo de Tabela de Conteúdo dos <i>trackers</i> para o domínio do <i>Tracker3</i>	82
7	Exemplo de Tabela de Conteúdo do <i>Root Tracker</i>	83
8	Dados de <i>trackers</i> BitTorrent públicos. Adaptado de: (ISOHUNT, 2012)	92
9	Comparação dos resultados dos testes: tempo de inicialização, tempo médio para a obtenção dos blocos e interrupção da apresentação . . .	137
10	Satisfação dos requisitos não funcionais	141
11	Satisfação dos requisitos funcionais	142

LISTA DE ACRÔNIMOS

API *Application Programming Interface*

ASPEN *Adaptative Spatial P2P Network*

CAN *Content Addressable Network*

CDN *Content Distribution Network*

CIA *Centralized Indexing Architecture*

CIDR *Classless Inter-Domain Routing*

CSV *Comma-Separated Values*

DAG *Directed Acyclic Graph*

DIFA *Distributed Indexing with Flooding Architecture*

DIHA *Distributed Indexing with Hashing Architecture*

DNS *Domain Name System*

DHT *Distributed Hash Tables*

EPG *Electronic Programming Guide*

FTP *File Transfer Protocol*

HTTP *HyperText Transfer Protocol*

ID *Identification Data*

IP *Internet Protocol*

JMF *Java Media Framework*

jVLC *Java bindings for VideoLan Client*

JXTA *Java Juxtapose*

LARC Laboratório de Arquitetura e Redes de Computadores

MIB *Management Information Base*

MMAPS *Market Management of Peer-to-Peer Services*

IP *Internet Protocol*

IPTV *Internet Protocol TeleVision*

IS-IS *Intermediate System to Intermediate System*

MA Métrica Analítica

MB Métrica Básica

MC Métrica Composta

ME Métrica Empírica

MGP Módulo Gerenciador de Perfis

NTP *Network Time Protocol*

OSPF *Open Shortest Path First*

P2P *Peer-to-Peer*

P2PM *Peer-to-Peer Managed*

QoE *Quality of Experience*

QoS *Quality of Service*

RFC *Request for Comments*

RTP *Real Time Protocol*

RTT *Round-Trip Time*

SLA *Service Level Agreement*

SMTP *Simple Mail Transport Protocol*

SNMP *Simple Network Management Protocol*

SPA *Shortest Path Algorithm*

SPF *Single Point of Failure*

TC Tabela de Conteúdo

UCP2P *User Centric P2P*

VLAN *Virtual Local Area Network*

VLC *VideoLan Client*

VoD *Video on Demand*

XML *Extensible Markup Language*

1 INTRODUÇÃO

O uso cada vez maior da Internet, aliado à demanda por aplicações que empregam cada vez mais recursos multimídia, criou um conjunto de requisitos para distribuição de conteúdo a ser satisfeito para obter determinadas características de desempenho e *Quality of Experience* (QoE) (PATHAN; BUYYA, 2007). Esses requisitos estão muitas vezes relacionados à necessidade de desempenho com base nos recursos de rede disponíveis, conduzindo à busca e ao desenvolvimento de novas abordagens para distribuição do conteúdo.

A distribuição de conteúdo multimídia através de Internet pode ter diferentes requisitos associados a padrões de *Quality of Service* (QoS) e QoE, que variam de acordo com o tipo de serviço oferecido. Entre os principais tipos de serviços de distribuição de conteúdo estão:

- Transmissão ao Vivo (*Live content*): que corresponde a conteúdo sendo ofertado ao mesmo tempo em que ele está sendo produzido ou captado. Exemplos: uma transmissão ao vivo de uma partida de futebol, de uma palestra ou de um curso; e
- Conteúdo sob Demanda (*On-demand content*): que é o conteúdo que já existe e está sendo disponibilizado para acesso. Exemplos: um arquivo executável de um programa ou um filme.

O paradigma clássico de distribuição de conteúdo, baseado na arquitetura cliente-

servidor, é de escalabilidade limitada devido à necessidade de garantir largura de banda e processamento suficientes para os múltiplos acessos simultâneos e concorrentes de milhões de usuários (NII, 1997). Como uma solução possível para esse problema são apresentados os sistemas *Peer-to-Peer* (P2P), que buscam explorar a largura de banda dos usuários (*peers*) e dados existentes localmente para auxiliar na distribuição do conteúdo de forma mais adequada. Porém, os *peers* não possuem a mesma confiabilidade de um servidor do modelo cliente-servidor, até mesmo por terem uma tendência de mudarem constantemente o estado de funcionamento (operacional para não operacional) por motivos diversos.

1.1 Motivação

A partir do ano 2000, surgiram diversas abordagens de redes P2P, cada uma com suas próprias características e limitações (CHEN et al., 2007). Entretanto, desde meados de 2003 (IPOQUE, 2007), o BitTorrent vem ganhando destaque pelos seus aspectos qualitativos (possibilidades de uso e recursos de gerenciamento) e quantitativo (quantidade de usuários no mundo que adotam o sistema). O fato de ser empregado tanto por usuários finais, que o utilizam com requisito de desempenho do tipo “melhor esforço”, quanto por empresas que demandam por parâmetros críticos de QoS, fazem com que o BitTorrent seja alvo de vários estudos sob diferentes aspectos (CHOE et al., 2007; ZHOU; DAI; LI, 2007; XIE et al., 2008). Diferentemente do compartilhamento entre usuários, no qual o desempenho nem sempre é uma fator determinante de escolha, o uso comercial de P2P por organizações requer um nível de desempenho da aplicação satisfatório para utilização do serviço e captação de clientes. Esse aspecto conduziu a pesquisas de otimização dos protocolos P2P, como por exemplo o uso de algoritmos de localidade, que é uma abordagem comumente empregada em trabalhos relacionados (KALOGERAKI; GUNOPULOS; ZEINALIPOUR-YAZTI, 2002; DARLAGIANNIS; MAUTHE; STEINMETZ, 2004; KOBAYASHI et al., 2005).

Algoritmos de localidade usam informações sobre o posicionamento dos nós da rede, bem como sobre os recursos disponíveis na mesma, para determinar um conjunto otimizado de *peers* para executar determinada tarefa, como fazer a transferência de algum conteúdo. Mais precisamente, tais algoritmos permitem à aplicação escolher um conjunto de *peers* para a distribuição de conteúdo de uma maneira mais meticulosa (ao invés de se basear simplesmente em mecanismos de seleção aleatória), o que garante uma melhor utilização dos recursos disponíveis e também um maior desempenho da rede P2P. Como benefício adicional, o conceito de localidade também pode ser usado para facilitar estratégias de armazenamento intermediário de conteúdos (*caching*), comumente usados em aplicações multimídia (Ernst-Desmulier et al., 2005); neste contexto, algoritmos de localidade podem ser empregados na determinação dos pontos de *caching* de uma forma mais eficiente e dinâmica. Devido ao uso de mecanismos elaborados, como os descritos anteriormente, as redes P2P são capazes de fornecer serviços com um desempenho adequado, apesar de ainda seguirem o paradigma do “melhor esforço” (QIU; SRIKANT, 2004). No entanto, e apesar do potencial deste conjunto de mecanismos e sua utilização nas mais variadas redes P2P, isto não levou à criação de um arcabouço de utilização e desenvolvimento que fosse padronizado e totalmente implementável (SHANAHAN; FREEDMAN, 2005). Por esta razão, diversas organizações acabaram desenvolvendo suas próprias soluções de localidade (e.g., (XIE et al., 2008; WARNER, 2012)), com o único objetivo de sanar suas necessidades particulares, uma tendência que também foi seguida pela comunidade acadêmica (HOWISON, 2003). Como resultado, cada nova proposta é projetada para aplicações, cenários e arquiteturas de rede específicas, adotando-se diversos mecanismos distintos para alcançar as suas metas. Na impossibilidade de avaliar todos os sistemas P2P, o presente trabalho é focado no BitTorrent por ser este o sistema P2P mais amplamente utilizado e pesquisado (IPOQUE, 2007; CAIDA, 2010) no momento do desenvolvimento desta tese.

1.2 Descrição do Problema

Um dos problemas mais comuns identificados em uma arquitetura semelhante ao BitTorrent é a falta de percepção de localidade (BARBOSA et al., 2004): quando um *peer* ingressa em um *swarm* BitTorrent¹, ele recebe uma lista contendo os *peers* que podem lhe fornecer o conteúdo; esta lista, entretanto, contém *peers* selecionados de forma aleatória (XIE et al., 2008; PAUL, 2008), incluindo *peers* que estão bastante distantes do requisitante a despeito da existência de alternativas mais próximas do mesmo.

Uma solução possível para esse problema é o emprego de uma base de dados com informações de geolocalização dos endereços IPs associados a cada *peer*, criando a percepção de localidade (CAI; WANG, 2006). Essa abordagem não é muito precisa; visto que a distância geográfica não implica necessariamente em distância em termos de redes de computadores que considera outros fatores, como por exemplo a latência entre os nós. Além disso, o grau de utilização dos recursos de rede não é levado em conta no processo de seleção de *peers*.

Outra limitação do BitTorrent é que as políticas padrão do protocolo para escolha de *peers*, dentre os participantes, na lista obtida visam essencialmente incentivar o compartilhamento do conteúdo e mantê-lo disponível o máximo de tempo possível na rede, de modo que os recursos da mesma são frequentemente empregados de um modo não otimizado (LEGOUT; URVOY-KELLER; MICHARDI, 2006; PAUL, 2008).

Finalmente, dado que um *tracker* BitTorrent² não possui informações sobre a distribuição das partes do conteúdo pelos *peers* no *swarm*, a lista de *peers* retornada pode incluir *peers* sem utilidade para o *peer* requisitante por não possuírem nenhuma das partes do conteúdo efetivamente desejadas. De fato, em um caso extremo, esta lista pode ser composta inteiramente por *peers* com esta característica (COHEN, 2008).

¹Denominação dada a um conjunto de *peers* que compartilha interesse por um determinado conteúdo. Maiores detalhes disponíveis na seção 2.3.

²Responsável por fazer um gerenciamento básico dos *peers* interessados por um conteúdo, maiores detalhes serão apresentados na seção 2.3.

Mesmo em casos menos drásticos, o esforço despendido na comunicação com esses *peers* não utilizáveis pode afetar a qualidade de serviços baseados em redes P2P do tipo BitTorrent.

1.3 Objetivos

Neste contexto, o objetivo desta tese é propor uma solução de uso de redes P2P semelhante ao BiTorrent para serviços de fornecimento de conteúdo de vídeo com necessidades específicas de desempenho, nesta tese estes estão relacionados a distribuição de conteúdo de vídeo sob demanda. Sistemas BitTorrent podem ser usados para distribuir uma ampla gama de conteúdo e estes podem ser consumidos de diferentes formas (P2PEDUCATION, 2012). Nesse sentido, foi escolhido um tipo de aplicação e conteúdo para orientar os requisitos de desempenho. Foi escolhido, como tipo de conteúdo, o vídeo por esse ser o tipo de conteúdo que gera a maior quantidade de tráfego na Internet (VESTAL, 2012; LABOVITZ, 2012; VEGA, 2012) e a aplicação *Internet Protocol TeleVision (IPTV)* foi escolhida por suas características de infraestrutura. Uma quantidade considerável de usuários (*peers*) e dados é um cenário favorável a sistemas P2P, visto que eles usualmente aumentam a sua eficiência proporcionalmente a quantidade de *peers*. A aplicação IPTV pode consumir o conteúdo de diversas formas (sob demanda - a medida que suas partes são obtidas ou reprodução - após todo o conteúdo ser adquirido), sendo escolhido o consumo sob demanda pela possibilidade de se utilizar parte da solução já desenvolvida por (GALLO, 2009). Com base nesse cenário é apresentada uma solução que leva em consideração: aspectos de localidade, os recursos da rede e as suas políticas de uso. Cabe salientar, que a solução proposta é voltada para provedores de conteúdo que utilizam suas próprias infraestruturas de redes para fornecer os seus serviços de conteúdo, e deste modo deve permitir que os mesmos realizem determinadas operações de gerenciamento sobre esses recursos.

1.4 Método

O método empregado para a execução deste trabalho foi a Pesquisa Aplicada baseada no método Hipotético-Dedutivo, utilizando referências científicas da área na definição do problema, especificação da hipótese de solução e avaliação da hipótese. A solução é especificada e implementada através de um protótipo de modo a gerar resultados que possibilitem a validação dessa hipótese.

Inicialmente é apresentado um estudo contextual sobre redes P2P objetivando identificar suas principais características, benefícios, problemas e abordagens. Esta contextualização visa a identificação dos conceitos básicos relacionados a P2P de modo a possibilitar o entendimento dos algoritmos de localidades aplicados nestas redes. Alguns dos principais algoritmos de localidade são brevemente explicados e organizados em categorias para fornecer a percepção das diferentes abordagens de uso dos conceitos de localidade, assim como os aspectos envolvidos. A classificação dos algoritmos de localidade possibilitou uma identificação mais precisa das características que são compatíveis com o problema descrito nesta tese, além de fornecer uma visão sobre as soluções que já empregam conceitos de localidade na distribuição de conteúdo em redes P2P. Então, esses mesmos critérios classificatórios são usados no processo de especificação dos requisitos da proposta de solução.

A partir das informações obtidas das soluções de localidade estudadas, e juntamente com a fundamentação de sistemas P2P, foi possível definir os requisitos de uma solução de distribuição de redes P2P que levasse em consideração: aspectos de localidade, os recursos da rede e suas políticas de uso. Os componentes desta solução foram selecionados com base nesses requisitos e então organizados de forma a satisfazê-los, definindo a hipótese de solução do problema.

Uma solução foi, então, definida (arquitetura usando *trackers* hierárquicos para localidade *Peer-to-Peer* em redes gerenciadas, aqui chamada de *Peer-to-Peer Manager*)

ged (P2PM)), podendo ser configurada de acordo com os requisitos do serviço que será oferecido através do sistema P2P. Partindo do princípio que os sistemas P2P são especificados para atender requisitos de aplicações específicas, isso tornou necessária a escolha de uma aplicação-exemplo para avaliar a solução proposta nessa tese. Desta forma, o presente trabalho usou como base a aplicação e o protótipo desenvolvidos na dissertação de mestrado de Diego S. Gallo, intitulada “Arquitetura de IPTV com Suporte à Apresentação Deslocada no Tempo Baseada em Distribuição Peer-to-Peer” (GALLO, 2009).

Em sua dissertação, (GALLO, 2009) propõe um sistema de distribuição de conteúdo de vídeos baseado em uma abordagem híbrida de *multicast* (transmissão ao vivo) e P2P (conteúdo sob demanda)³ (GALLO et al., 2008). A distribuição P2P é baseada em sistemas do tipo BitTorrent e toda a proposta tem como plataforma a rede privada de um provedor de conteúdo. Sendo assim, esse sistema alia os desafios de desempenho dos sistemas P2P ao das aplicações de vídeo, que necessitam de elevado desempenho para fornecer uma QoE adequada e a QoS (GREENGRASS; EVANS; BEGEN, 2009). Nesse sentido, o seu uso para verificar a eficiência de algoritmos de localidade propostos adequa-se aos objetivos desta tese. Outro motivo para a utilização do sistema proposto (GALLO, 2009) foi o acesso ao código do sistema e o conhecimento do autor sobre o mesmo, o que facilitou a adição de novas funcionalidades no sistema e a utilização de uma versão adaptada do protocolo BitTorrent.

O uso do serviço de vídeo (como aplicação que a rede P2P suportará) auxilia na definição de valores e condições para avaliar a eficiência da proposta, enquanto os aspectos operacionais de um sistema BitTorrent auxiliam na definição de limites mínimos de operação do sistema P2P. Fundamentado nessas informações, foram definidos os casos de uso implementados e a natureza dos testes realizados. Os experimentos relacionados a uma das etapas finais da Pesquisa Aplicada foram realizados em um

³Essa arquitetura é descrita com mais detalhes no Capítulo 5.

ambiente de testes (*testbed*) com a instalação do protótipo especificamente preparado para essa finalidade, com base nos casos de uso.

A análise dos resultados completou o método empregado neste trabalho, fornecendo uma avaliação sobre o quanto factível é a hipótese formulada ao analisar os resultados coletados nos experimentos, confrontados com os requisitos do serviço oferecido através da aplicação baseada em P2P. Esta análise também possibilitou verificar o comportamento da solução como um todo e definir pontos de interesse para futuras pesquisas. Por fim, cabe ressaltar que toda a pesquisa foi desenvolvida dentro do LARC (Laboratório de Arquitetura e Redes de Computadores) do PCS (Departamento de Engenharia da Computação e Sistemas Digitais) na EPUSP (Escola Politécnica da Universidade de São Paulo).

1.5 Organização do trabalho

O presente trabalho está organizado em capítulos e seções autocontidos, que possibilitam ao leitor com conhecimentos sobre determinado assunto ir diretamente aos tópicos de seu interesse. Após o capítulo introdutório, os Capítulos 2 e 3 fornecem ao leitor os conhecimentos básicos necessários para compreender a proposta de solução apresentada nessa tese. Basicamente, são contextualizadas as redes P2P e os algoritmos de localidade aplicáveis a tais redes.

O Capítulo 4 descreve os pressupostos e requisitos que serão usados como delimitadores na especificação da solução proposta, chamada de P2PM. Como solução, é especificada uma arquitetura usando *trackers* hierárquicos para localidade em redes P2P gerenciadas, sendo descrita a motivação da escolha dos elementos envolvidos nessa solução. Também são definidos alguns casos de uso, que devem ser aplicáveis no cenário em que esta solução for empregada.

O P2PM, descrita no Capítulo 4, é aplicado a um serviço de IPTV oferecido atra-

vés de redes P2P. No Capítulo 5 é apresentada a descrição do sistema e do cenário onde os experimentos são realizados, bem como a definição dos requisitos e critérios de desempenho que devem ser satisfeitos para o sistema ser considerado satisfatório. Os experimentos são realizados usando uma aplicação de IPTV baseada na solução desenvolvida por (GALLO, 2009), na qual foram realizadas diversas modificações e adicionados novos recursos.

Uma vez definidos o sistema e experimentos é apresentada uma análise dos resultados no Capítulo 6. Os resultados relatados são referentes ao protótipo desenvolvido para fornecer o serviço de vídeo sobre P2P, empregando métricas de desempenho pertinentes a sistemas desse tipo. Por fim, são apresentadas as considerações finais, onde são relatadas as contribuições desse trabalho e os trabalhos futuros que podem ser desenvolvidos a partir dos resultados obtidos nessa tese.

Adicionalmente, foram incluídos três apêndices:

- Apêndice A - Publicações. Lista com todas as publicações realizadas pelo proponente desta tese, durante o período de doutoramento, tanto as diretamente relacionadas com o tema principal da tese como as não diretamente relacionadas.
- Apêndice B - Arquivo XML dos perfis dos grafos. Inclui uma listagem dos arquivos dos perfis dos grafos que foram empregados para executar os experimentos descritos no Capítulo 6.
- Apêndice C - Interface de monitoração em tempo real. Apresenta, em figuras explicativas, algumas telas da interface de monitoração disponível na solução P2PM.

2 CONCEITOS BÁSICOS

Um dos fatores importantes que motivam os estudos sobre redes P2P é a simplicidade com que as mesmas podem ser montadas; outro é a igualdade dos computadores da rede no que se refere às suas potencialidades funcionais. No entanto, os computadores empregados como *peers* podem ter diferentes capacidades de processamento, de armazenamento e largura de banda de seus enlaces, de modo que suas potencialidades de desempenho são diferentes (FLENNER, 2003). Ao mesmo tempo que os usuários (*peers*) não têm a obrigatoriedade de serem permanentes, podendo retirar-se da rede a qualquer hora por qualquer motivo, a organização da rede pode ser bastante dinâmica e imprevisível também.

Devido à facilidade de construção, é possível a elaboração de vários cenários de redes P2P. Cada cenário normalmente é moldado para otimizar determinadas características das aplicações que serão executadas sobre essa rede. Em (HAMRA; FELBER, 2005) é apresentado um estudo específico sobre as opções de projeto de organização que podem ser empregadas em redes de distribuição baseadas em P2P (exemplos incluem modelos baseados em malhas e árvores), levando em consideração a quantidade de camadas e operações típicas de tais redes.

2.1 Redes P2P

Segundo (ORAM, 2001; LI, 2006), o termo P2P refere-se às redes de computadores nas quais espera-se que os usuários que a integram contribuam com seus arquivos,

poder de processamento e outros recursos para execução de diferentes tarefas. Um dos aspectos mais interessantes sobre esses sistemas é o seu “potencial perturbador social”, pois a maioria das opções de controle da rede estão sob a responsabilidade de usuários comuns e não sob a responsabilidade de um servidor confiável como no modelo cliente-servidor. A Figura 1 ilustra as diferenças entre os modelos cliente-servidor e distribuído(P2P).

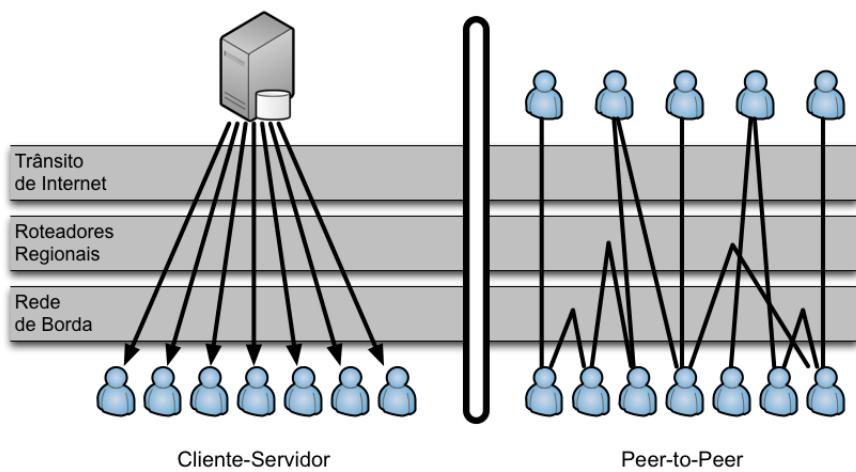


Figura 1: Modelo centralizado cliente-servidor e modelo distribuído P2P

Analisando a Figura 1, pode-se notar que para obter um conteúdo no modelo cliente-servidor é necessário que cada usuário percorra os diversos níveis de roteadores de distribuição (Trânsito Internet / WAN, Roteadores Regionais e Rede de Borda). Em uma abordagem simplista, isto faz com que o mesmo conteúdo percorra n vezes todos os níveis para que n clientes sejam atendidos, provocando sobrecarga na rede e comprometendo a escalabilidade desse modelo (principalmente devido aos requisitos de largura de banda). Um sistema P2P nessa mesma situação irá explorar as cópias inteiras ou partes do conteúdo que já foram obtidas por outros *peers*, de modo a evitar que o conteúdo percorra todos os níveis cada vez que um *peer* o solicita.

Conclui-se então que um sistema P2P é basicamente uma conexão direta entre dois ou mais usuários(*peers*) que estejam dispostos a trabalhar em conjunto, na troca de qualquer tipo de arquivo ou programa de computador, onde as duas ou mais partes têm alguma capacidade de comunicação.

As principais características das redes P2P podem ser resumidas como:

- **São auto-organizáveis.** No caso das redes P2P, pode-se partir da premissa que não existe nenhum elemento organizador dos *peers* e, deste modo, os *peers* interagem entre si e organizam-se de diversas formas para realizar seus serviços (LOGUINOV; CASAS; WANG, 2005).
- **Suporam o compartilhamento de recursos** (processador, sistema de arquivos, etc.), conforme a aplicação. O compartilhamento pode ser configurado pelos *peers*, que podem estabelecer limites de acordo com a sua vontade/capacidade, ou então gerenciado por uma aplicação, como é o caso de algumas redes P2P estruturadas (CEBALLOS; GORRICO, 2006).
- **São baseadas na colaboração voluntária.** O fato dos *peers* serem usuários na extremidades da rede (Rede de Borda) implica que os mesmos não possuem obrigação explícita de participar da rede. A participação é motivada normalmente pelo interesse em obter recursos ou disponibilizá-los (STUTZBACH; REJAIE, 2006).
- **Os *peers* que compõem o sistema P2P são funcionalmente iguais.** Tal propriedade é reforçada pelo fato que a aplicação P2P executada em cada *peer* é a mesma. No entanto, tal equivalência funcional não significa necessariamente que todos os *peers* irão ter as mesmas características ou desempenho (ORAM, 2001; LI, 2006).
- **São, geralmente, compostas por uma grande quantidade de *peers* conectados em rede.** Se uma rede possuir apenas dois *peers*, ela acaba tornando-se conceitualmente um sistema cliente-servidor, porém sem a confiabilidade deste. Para uma rede P2P ser operacional é necessário que ela seja formada por uma grande quantidade de *peers* que possam compartilhar seus recursos (VU et al., 2007).

- **Exploram os recursos da borda de uma rede de computadores (Rede de Borda).** A fim de aproveitar os recursos de rede locais (como largura de banda por exemplo) e evitar que os dados percorram muitos níveis na rede, sistemas P2P buscam encontrar *peers* próximos e selecionar os que oferecem os recursos mais apropriados para atender às suas necessidades (XUE; YOU; JIA, 2004).
- ***Peers* possuem conectividade intermitente.** Devido à colaboração dos *peers* ser voluntária, não existe uma ordem definida que determine quando e por quanto tempo que permanecerão na rede, muito menos quais recursos que irão compartilhar (LEONARD; RAI; LOGUINOV, 2005; STUTZBACH; REJAIE, 2006).

Uma outra característica das redes P2P é que elas criam uma nova rede lógica sobre uma rede física de computadores, que é conhecida como rede de sobreposição ou simplesmente *overlay*. A rede *overlay* define como logicamente os *peers* estão conectados. Além destas características, as redes P2P podem ser classificadas de acordo com três modelos principais de estrutura e funcionamento (ORAM, 2001):

- **CIA (*Centralized Indexing Architecture*).** As redes P2P mantêm um sistema de busca centralizado em um servidor;
- **DIFA (*Distributed Indexing with Flooding Architecture*).** Nessas redes as buscas não são centralizadas em um determinado número de servidores, seu sistema de busca não segue uma sequência organizada de varredura (não estruturado); e
- **DIHA (*Distributed Indexing with Hashing Architecture*).** Este modelo de rede P2P não possui um ponto centralizador e o sistema de busca é estruturado (normalmente através de *hashes*), fazendo assim com que as buscas por um determinado recurso tenham maior chance de sucesso.

Além desses modelos, as redes P2P também podem ser classificadas pelo (BOUTABA, 2008):

- **Grau de descentralização P2P.** Essa dimensão identifica a dependência ou não de *peers* especiais (que realizam algumas atividades extras que não são executadas por todos os demais *peers*) ou o emprego de servidores para coordenar o funcionamento da rede P2P. Podem-se ter, então:
 - P2P descentralizado híbrido. Existe um servidor central que facilita a interação entre os *peers*. Normalmente esse servidor central realiza a busca e a identificação de *peers na rede*. Exemplo: Napster¹;
 - P2P parcialmente centralizado. Alguns *peers* podem desempenhar atividades extras para operacionalizar a indexação dos *peers* na rede. Esses *peers* especiais, também conhecidos como super-nós ou *super-peers*, usualmente são responsáveis por uma parte da rede P2P na qual eles identificam os *peers* e fornecem informações sobre os respectivos recursos compartilhados. Exemplo: KaZaa²; e
 - P2P descentralizado puro. Todos os nós realizam exatamente as mesmas tarefas e não há uma coordenação de atividades centralizada. Exemplo: GNUTella³ original e Freenet⁴.
- **Grau da estrutura P2P.** Essa dimensão identifica a organização entre os *peers* e demais elementos da rede P2P para estruturação de uma rede *overlay*. Sendo subdividida em:
 - P2P estruturado. A topologia da rede *overlay* é rigidamente controlada e os recursos alocados em locais específicos da rede. É comum, neste tipo de rede, ser fornecido um mapeamento entre o identificador do conteúdo e sua localização. Exemplos: Chord (STOICA et al., 2001), CAN (RATNASAMY et al., 2001) e Tapestry (ZHAO; KUBIATOWICZ; JOSEPH, 2001);

¹www.napster.com

²www.kazaa.com

³rfc-gnutella.sourceforge.net/

⁴freenetproject.org/

- P2P não estruturado. Os recursos estão distribuídos aleatoriamente pelos *peers* e são empregados mecanismos de *broadcast* para realizar a busca. A alocação dos recursos não possui qualquer relação com a topologia da rede *overlay*. Exemplos: Napster, GNUTella e KaZaa; e
- P2P fracamente estruturado. Esse tipo é um meio termo entre P2P estruturado e P2P não estruturado. A localização dos recursos é definida com noções (dicas) da rota até o *peer* que o detém, porém a localização exata não é especificada. Exemplo: Freenet.

A Tabela 1 resume a classificação definida por (BOUTABA, 2008) e enquadra os exemplos de redes P2P citadas na explicação dos itens anteriores:

Tabela 1: Classificação das Redes P2P. Adaptado de (BOUTABA, 2008)

	P2P não estruturado	P2P fracamente estruturado	P2P estruturado
P2P descentralizado híbrido	Napster	BitTorrent	-
P2P descentralizado puro	GNUTella	Freenet	Chord, CAN e Tapestry
P2P parcialmente centralizado	KaZaa	-	-

As diversas características das redes P2P são em boa parte uma consequência natural das diversas aplicações desenvolvidas para criar redes P2P, e em outra parte fruto das pesquisas acadêmicas para promover o melhor uso do conceito de redes P2P. Deste modo, e para melhor entendimento destes fatos, é pertinente o estudo do histórico das principais aplicações P2P.

2.2 Histórico das redes P2P

Antes do P2P popularizar-se, no ano de 2000 foi criada uma das primeiras redes P2P conceituais: a Freenet, desenvolvida pelo pesquisador Ian Clarke da Universidade de Edinburgo. A Freenet implementa um meio simples de troca de arquivos que provou o bom funcionamento desta funcionalidade entre computadores, o que constitui a

essência das redes P2P (ORAM, 2001).

Segundo (HANDURUKANDE et al., 2006), os sistemas P2P tornaram-se mundialmente conhecidos no ano de 2000 com a popularização do Napster, criado em 1999 para permitir o compartilhamento de arquivos de música no formato MP3. Apesar das questões legais envolvidas, em função de músicas compartilhadas sem o consentimento dos autores, um ponto interessante sobre estes sistemas P2P é que os mesmos foram projetados para oferecer características de escalabilidade e eficiência na distribuição de arquivos. Após o Napster, surgiram outros sistemas com o mesmo propósito, como o GNUTella no ano de 2000 e KaZaa em 2001. As análises iniciais sobre estes sistemas revelaram o fato de que nem todos os *peers* colaboravam efetivamente na distribuição do conteúdo. Na verdade, vários *peers* somente obtinham conteúdo sem depois compartilhá-lo, sendo chamados de *free-riders* devido a esse comportamento egoísta (CEBALLOS; GORRICHÓ, 2006).

Para minimizar o problema gerado pelos *free-riders*, as aplicações da geração P2P seguinte, como BitTorrent (2001) e eMule (2002), criaram mecanismos que incentivam os *peers* a compartilharem conteúdo para conseguirem obter o arquivo inteiro, prevendo o surgimento de redes P2P compostas por diversos *free-riders* que obtinham conteúdos de alguns poucos *peers* altruístas.

O projeto *Java Juxtapose* (JXTA), criado no ano de 2001 e baseado na plataforma Java, também é relevante na história das redes P2P, pois disponibilizou uma plataforma para construção de aplicações P2P. A ideia por trás do projeto é que as redes P2P podem ser exploradas de maneira mais rápida usando objetos prontos e assim permitir que o desenvolvedor preocupe-se mais em fazer as funcionalidades da aplicação e menos em desenvolver os mecanismos de P2P (HALEPOVIC; DETERS, 2005).

Observando as principais aplicações e redes P2P constata-se que a motivação no seu uso por parte dos usuários refere-se principalmente ao compartilhamento de arquivos na Internet e pelo desempenho obtido neste processo. A Figura 2 mostra um

gráfico dos principais tipos de protocolos de aplicação que trafegaram pelos principais *backbones* da Internet no EUA no período de 1993 a 2006.

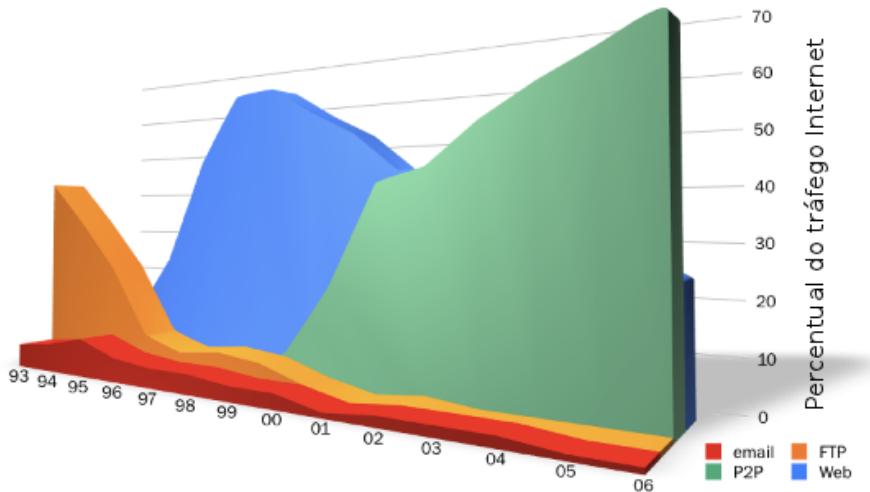


Figura 2: Estatística do tipo de tráfego nos principais *backbones* de rede dos EUA.
Fonte: (IPOQUE, 2007)

Analisando a Figura 2, percebe-se o crescimento da proporção do tráfego P2P no total de dados trafegados na Internet a partir do ano de 2000, quando surgiu o Napster, e seu forte crescimento nos anos seguintes. Percebe-se que à medida que o percentual de tráfego P2P cresce o percentual de tráfego *web* diminui, isto é, os usuários perceberam um bom desempenho na obtenção de conteúdo usando o modelo P2P quando comparado ao modelo cliente-servidor tradicional sobre o qual a *web* é construída. Também é fato que as *Content Distribution Network* (CDN) nesse período estavam restritas essencialmente à distribuição de softwares e possuíam um custo considerável de manutenção. O crescimento do percentual do tráfego P2P foi uma consequência direta do tipo de arquivos compartilhados: vídeos e programas diversos (legais e pirataria). O percentual de tráfego *web* (HTTP) somente ultrapassou o de P2P (ano 2003) devido ao acesso de conteúdos fornecidos por *sites* de vídeo como o Youtube e Flash, o que somente foi possível através da estruturação de CDN. Além disso, as CDN começaram a se tornar mais acessíveis e os serviços de vídeo como o Youtube passaram a disponibilizar gratuitamente para os usuários uma ampla gama de conteúdos via

tecnologias do tipo Flash que entregam o conteúdo via protocolo *HyperText Transfer Protocol* (HTTP). Esse fato é analisado por (ERMAN et al., 2007) que identifica o tipo de conteúdo disponibilizado e a facilidade de acesso como fatores determinantes para a migração do volume de tráfego da Internet entre protocolos. Já no que diz respeito à proporção de tráfego P2P, a maior parte é gerada por aplicações que utilizam o protocolo BitTorrent. A Figura 3 mostra que entre 2008 e 2010 essa tendência de aumento do tráfego *web* continua, ainda motivada pelo conteúdo de vídeo sobre HTTP (CAIDA, 2010).

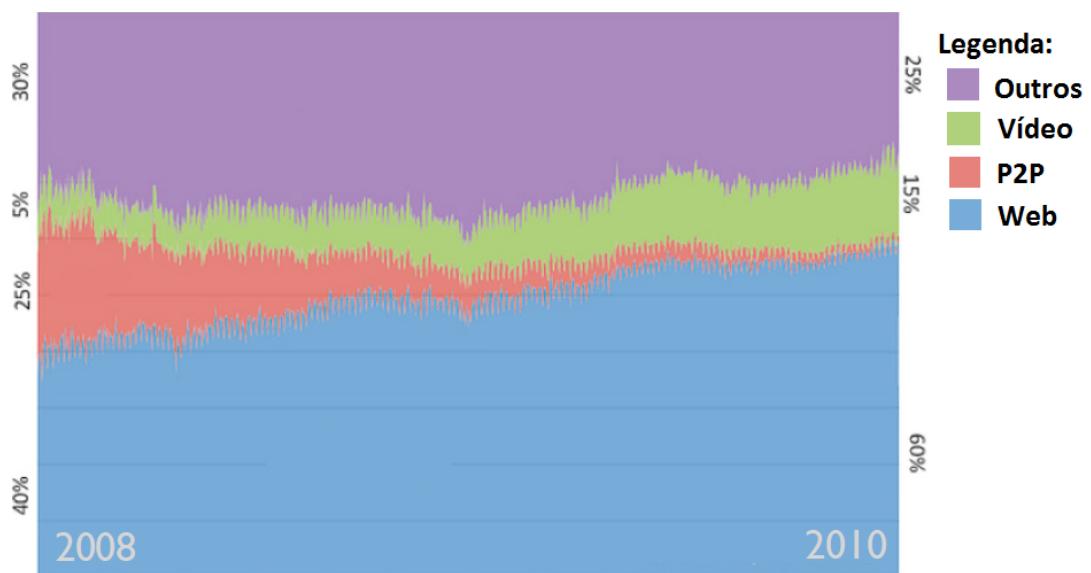


Figura 3: Composição do tráfego agregado na Internet no período entre os anos de 2008 a 2010. Fonte:(CAIDA, 2010).

A Figura 4 mostra o percentual de tráfego atual na América do Norte gerado por acesso fixo, observa-se que o *streaming* tem apresentado um crescimento contínuo nos últimos anos (SANDVINE, 2012).

O volume de tráfego P2P continua a representar uma parcela significativa do tráfego na Internet. Tal fato indica a demanda por parte dos usuários de conteúdo multimídia que ainda não é disponibilizado através de CDN. Entretanto, a QoS desse serviço é baseada no modelo de "melhor esforço", no qual nem sempre os usuários conseguem o que desejam com um desempenho satisfatório (CHO et al., 2006).

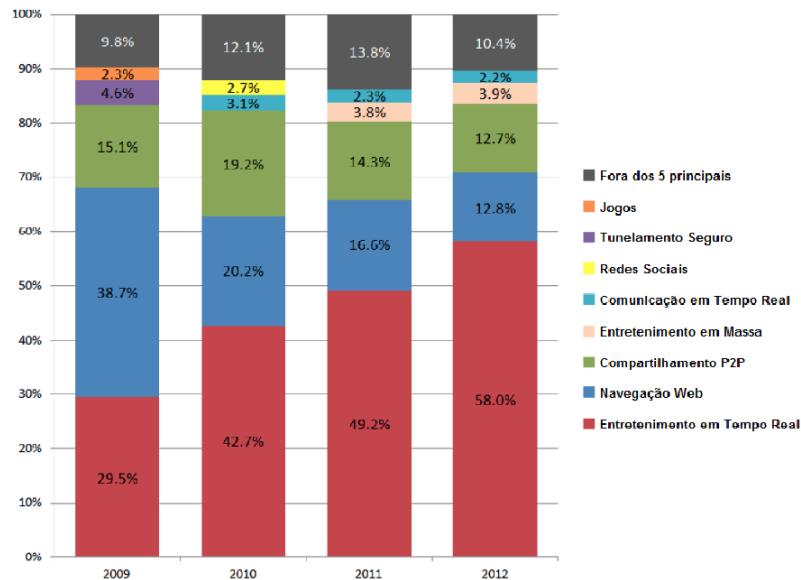


Figura 4: Composição do tráfego agregado na Internet na América do Norte entre os anos de 2009 a 2012. Fonte:(SANDVINE, 2012).

Empresas de entretenimento como a Warner Bros e outras (XIE et al., 2008; WARNER, 2012) estão começando a disponibilizar conteúdo multimídia na Internet e a comercialização do conteúdo através dos sistemas de distribuição baseado em P2P BitTorrent é um caminho natural diante das limitações de distribuição de conteúdo via o modelo cliente-servidor. Logo, entender como o BitTorrent funciona auxilia a descobrir formas de otimizá-lo e mensurá-lo (GUO et al., 2005; DAS; KANGASHARJU, 2006).

A Inserção de tráfego de P2P comercial traz requisitos de QoS nos quais a confiabilidade e disponibilidade são métricas básicas, exigindo uma abordagem alternativa ao "melhor-esforço". A preferência de sistemas P2P (BitTorrent) explorando sua flexibilidade de uso e recursos de configuração, faz dele foco de estudos cada vez mais intensos na última década (QIU; SRIKANT, 2004; CHEHAI; XIANLIANG; HANCONG, 2006; NEGLIA et al., 2007).

2.3 Sistemas P2P BitTorrent

O BitTorrent é o resultado de cinco anos de pesquisas intensas, que resultaram em um software e um protocolo de comunicação para fazer a distribuição cooperativa de conteúdo (*On demand*) entre usuários/*peers*. Um dos fatores chave para o sucesso do

modelo P2P proposto foi a política de “olho-por-olho” ou “dar para receber”, que se explica como: os *peers* só conseguem obter conteúdo se também compartilharem esse mesmo conteúdo com outros *peers*; assim os *peers* que adotam a postura de *free-riders* acabam sendo penalizados em decorrência dos algoritmos empregados. Desse modo é promovida o que se chama de distribuição cooperativa (COHEN, 2003).

A distribuição cooperativa é um modelo que permite o crescimento de nós na rede de uma maneira quase ilimitada, porque cada novo *peer* que entra na rede não apenas buscará mas também fornecerá conteúdo, aumentando a oferta de partes do mesmo conteúdo que é buscado. Pode-se dizer então que a popularidade de um conteúdo cria um “círculo virtuoso” porque cada novo participante traz novos recursos de distribuição, aumentando a escalabilidade do sistema.

A especificação do protocolo BitTorrent de maneira pública proporcionou a sua rápida adoção tanto por usuários como por instituições. Com isso surgiram vários softwares para usar o sistema e também para disponibilizá-lo (COHEN, 2008).

2.3.1 Componentes de um Sistema BitTorrent

Um sistema BitTorrent é composto de (COHEN, 2003):

- **Arquivo .torrent:** Relativo ao conteúdo/arquivo que será disponibilizado através do sistema P2P BitTorrent. Contém informações estáticas necessárias para iniciar o processo de compartilhamento, como nome, tamanho, *checksum* e endereço IP do *tracker*. Exemplo: Kubuntu 12.10.torrent;
- **Repositório de Torrents:** Tipicamente, é implementado por um servidor *web* que armazena os arquivos de meta-informações .torrent acessíveis via HTTP. Este servidor *web* funciona como um repositório para que usuários possam encontrar o conteúdo desejado e obter o arquivo .torrent associado. Nota-se, entretanto, que os arquivos .torrent poderiam ser disponibilizados por outros meios

como *File Transfer Protocol* (FTP), e-mail ou até mesmo mídias removíveis.

Exemplo: <http://www.mininova.org>;

- **Tracker:** É um computador que não compartilha conteúdo, mas é empregado para auxiliar na coordenação dos *peers*. Cada *peer* que deseja obter um arquivo obtém o endereço do *tracker* associado no arquivo .torrent, o qual coordena os *peers* envolvidos no compartilhamento do conteúdo especificado no arquivo .torrent. Exemplo: <http://linuxtracker.org:27100>;
- **Peers:** São os nós que estão executando uma aplicação BitTorrent (software ou *plug-in* de navegadores *web*). Dependendo de sua situação na rede, eles podem ser classificados como:
 - **Downloader** ou *leecher*: não possui a totalidade do conteúdo associado ao .torrent;
 - **Seeder**: possui o conteúdo inteiro (completo), e escolhe ficar no sistema para que outros *peers* possam obtê-lo. Um *seeder* pode ser o primeiro computador a disponibilizar um conteúdo ou um *leecher* que obteve todo o conteúdo associado a um .torrent; e
 - **Swarm**: o conjunto de *peers* envolvido no compartilhamento de um conteúdo é denominado “swarm” (“enxame”). Os *swarms* são mantidos pelo *tracker* que usa esse conjunto para fornecer listas de *peers* para obtenção de conteúdo por parte dos *leechers*.

Percebe-se que um sistema BitTorrent é uma rede P2P estruturada no qual o *tracker* tem uma função coordenadora e os *peers* podem assumir simultaneamente os papéis de *leecher* e *seeder*, desde que a aplicação em execução no *peer* suporte múltiplos torrents. Para funcionar, um sistema BitTorrent necessita um *tracker* para coordenar os *peers* e que os *peers* possuam um arquivo .torrent com conteúdo coordenado por este *tracker*.

2.3.2 Etapas de Obtenção de Conteúdo

A Figura 5 ilustra o funcionamento básico de um sistema BitTorrent.

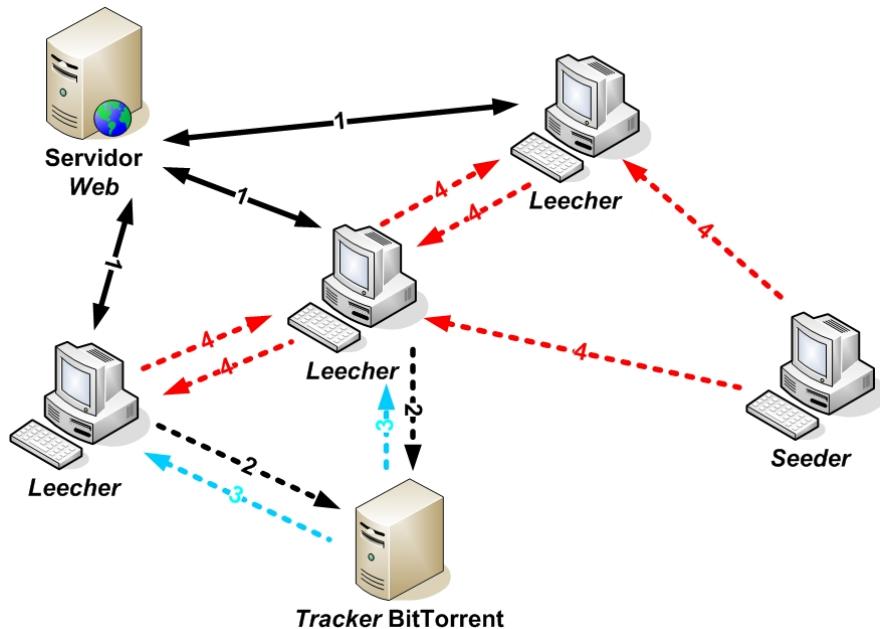


Figura 5: Funcionamento de um sistema BitTorrent básico

A Figura 5 identifica as principais etapas envolvidas na obtenção de conteúdo em um sistema BitTorrent, que são:

1. Um *Peer* busca e obtém um arquivo .torrent em um repositório de torrents;
2. O *peer* assume o papel de *leecher*, contacta o *tracker* para obter o endereço de outros *peers* e junta-se ao *swarm* relativo ao .torrent;
3. O *tracker* fornece a lista de *peers* que estão no *swarm* desse .torrent e inclui o novo *peer* nesse *swarm*; e
4. Transmissão de pedaços do conteúdo ou troca de *peers*, segundo o algoritmo.

A partir do momento que é iniciado este processo de distribuição cooperativa, os *peers* começam a fornecer/receber conteúdo. Este conteúdo é formado por um ou mais

arquivos que são divididos em pedaços de tamanho fixo, normalmente de 256Kb. Cada pedaço por sua vez é dividido em subpedaços para *download*, normalmente de 16kb, porém os *peers* somente fazem *uploads* de pedaços completos (já foram obtidos todos os subpedaços que compõem o pedaço). Desse modo, cada *leecher* informa para todos os *peers* do *swarm* quais os pedaços que ele possui, permitindo assim a trocar pedaços ou subpedaços entre eles. A verificação da integridade de cada pedaço é feita através da verificação/comparação do *hash* dos dados, sendo que os valores dos *hashes* de cada pedaço estão inclusos no arquivos .torrent (COHEN, 2003; COHEN, 2008).

2.3.3 Critérios de Seleção de Pedaços

A seleção de qual pedaço será obtido pelo *peer* segue um algoritmo de seleção que objetiva priorizar os *peers* que são cooperativos (fornecem e obtém conteúdo), assim como manter o conteúdo completo disponível na rede. Mais precisamente, os esquemas de seleção de pedaços são (COHEN, 2003; COHEN, 2008):

- **Prioridade estrita:** como primeira política de seleção é definido que, uma vez que o *peer* iniciou o *download* de um pedaço, é dada a prioridade para a obtenção de todos os demais subpedaços desse pedaço. O objetivo é obter pedaços inteiros para aumentar a cooperação, visto que o *peer* só compartilha os pedaços completos que ele possui.
- **Raros primeiro:** o pedaço a fazer o *download* é o que menos aparece na lista dos outros *peers*. Esta técnica tem como objetivo garantir que os *peers* tenham todos os pedaços que precisam, assim podem fazer o *upload* quando for necessário. Ela também assegura que os pedaços mais comuns tenham a sua obtenção menos priorizada, pois a probabilidade de que estes pedaços continuem disponíveis depois que outros pedaços mais raros foram obtidos é bem maior.
- **Primeiro pedaço aleatório:** Uma exceção para o mecanismo de “raros pri-

meiro” é quando o processo de obtenção do conteúdo é iniciado. Neste momento, é mais importante que o *peer* consiga um pedaço completo o mais rápido possível para que possa iniciar a disponibilizá-lo para os outros *peers* no *swarm*. Pedaços raros estão disponíveis em poucos *peers*, o que faz com que sua obtenção seja mais lenta e, portanto, retarde o ingresso do *peer* na cooperação da distribuição do conteúdo. Por este motivo, os pedaços são selecionados aleatoriamente até que um pedaço inteiro seja obtido e, então, muda-se a estratégia para “raros primeiro”.

- **Modo “fim de jogo”:** Algumas vezes um pedaço pode ser solicitado para um *peer* com baixa taxa de transferência, o que não é um problema quando se está no meio do processo, mas que pode tornar mais demorada a obtenção de todo o conteúdo. Neste modo o *peer* envia uma requisição de todos os seus subpedaços faltantes para todos os *peers* da sua lista em um esforço de completar os pedaços e iniciar o compartilhamento. Para evitar sobrecarregas da rede com subpedaços duplicados, são enviadas mensagens de cancelamentos à medida que os subpedaços são recebidos.

2.3.4 Critérios de Seleção de um Peer

Tendo analisado a forma de obtenção de conteúdo, é relevante compreender como são selecionados os *peers* correspondentes dentro de um *swarm* e como funciona a dinâmica de troca dos *peers*. Para tanto, são empregados os seguintes mecanismos internos de incentivo e seleção (COHEN, 2003; COHEN, 2008; EVANGELISTA et al., 2011):

- **Algoritmo de *Choking*:** O *choking* consiste no bloqueio temporário das requisições de *upload* por parte de um *peer*. A avaliação de *choking* é feita a cada 10 segundos e é baseada exclusivamente na taxa de *download* que possui atualmente com cada *peer* que está transferindo conteúdo, a qual é reavaliada a cada 20 segundos; com base nessas informações é decidido quais serão os *peers* que

serão desbloqueados ou bloqueados (*un/choked*). Cada *peer* comunica-se (*unchoke*) com um número fixo de *peers*, sendo que o número padrão é 4 (quatro).

- ***Unchoking Otimista***: Cada *peer* seleciona um outro *peer* aleatoriamente, independente das suas taxas de transmissão a fim de verificar se existem conexões com taxas de transferência ou pedaços melhores dos que os *peers* atuais. Como o Algoritmo de *Choking* também será executado sobre este *peer* no futuro, caso ele tenha bom desempenho/resultado ele pode tornar-se um *peer* efetivo da lista, substituindo o pior elemento da lista corrente de *peers*. Este mecanismo é normalmente executado a cada 30 segundos.
- ***Anti-snubbing***: Assume-se que um *peer* foi *choked* pelos outros *peers* quando não recebe dados em um intervalo contínuo de 60 segundos. Sendo assim, os outros *peers* não irão fazer *upload* para este *peer* a não ser pelo *Unckoking Otimista*. Esta situação pode ocorrer quando o *peer* tiver baixa taxa de transferência ou não estiver cooperando e é uma consequência do Algoritmo de *Choking* para evitar *free-riders*.

Observando os algoritmos empregados no sistema BitTorrent, percebe-se que a Figura 5 mostra o funcionamento deste sistema apenas em pequena escala. Atualmente, existem milhares de *trackers* dispersos pelo mundo e muitos outros milhares de *peers* empregando a distribuição cooperativa (ISOHUNT, 2012). Neste contexto, e apesar de uma política implícita de melhorar o esforço nos seus algoritmos, é difícil mensurar a eficiência dos mecanismos empregados em decorrência da variedade das taxas de transferência e largura de banda de cada rede onde está cada *peer*.

O cenário e arquitetura de uma rede BitTorrent padrão são feitos para estimular a cooperação entre os *peers*, sem levar em consideração que uma terceira aplicação pode estar consumindo o conteúdo obtido ou que a obtenção do conteúdo deve atender a determinadas características de desempenho/QoS. O comportamento dos sistemas

BitTorrent padrão é típico de uma política de “melhor esforço”.

2.3.5 Problemas nos Sistemas BitTorrent

Apesar do BitTorrent eliminar gargalos no tráfego dos dados, ele continua tendo um ponto central de falha: o *tracker*. A primeira geração de sistemas BitTorrent emprega um *tracker* centralizado, que permite a coordenação entre os *peers*. Caso o *tracker* torne-se inacessível, o sistema ficará indisponível para novos *peers*, pois eles não conseguirão obter o conteúdo informado no torrent e consequentemente não contribuirão com recursos para o sistema. Ao mesmo tempo, os *peers* que já estão no sistema (i.e., que já obtiveram uma lista de *peers*) dependerão exclusivamente da lista corrente para obter todas as partes do torrent em questão e, caso isto seja insuficiente, deverão ficar aguardando o *tracker* voltar a funcionar. Além deste problema específico, o BitTorrent também está sujeito aos problemas clássicos de sistemas P2P, como:

- ***Churn***: Fluxo de entrada e saída de peers no sistema, o que dificulta a localização e distribuição de conteúdo (BINZENHÖFER; LEIBNITZ, 2007);
- ***Free-riders***: Os *peers* que somente fazem *download* de conteúdo e não cooperam no sistema podem degradar o desempenho da rede P2P, algo evitado (porém não eliminado) por mecanismos como o *Choking* (COHEN, 2008; KARAKAYA; KÖRPEOGLU; ULUSOY, 2008);
- ***Flash-crowd***: Um *peer* ou rede muda repentinamente para um estado sobre-carregado de tráfego, prejudicando o compartilhamento de recursos (STADING; MANIATIS; BAKER, 2002; RUBENSTEIN; SAHU, 2005); e
- ***Hot-Spot***: Um *peer* que fica sobrecarregado de requisições, perdendo a capacidade de aproveitar as características distribuídas do sistema P2P (YU; LEE; ZHANG, 2005; NORROS et al., 2006). Enquanto o *flash-crowd* é um acontecimento

eventual menos frequente e que podem acontecer com *peers* diversos, o *hot-spot* é um evento mais frequente e que acontece com os mesmos *peers* repetidamente.

Os problemas identificados podem causar indisponibilidade do sistema BitTorrent ou fazer com que seu desempenho não atenda aos requisitos necessários da aplicação. Portanto, tais problemas devem ser levados em consideração quando for realizada a avaliação ou elaboração de um sistema baseado em BitTorrent, sendo necessários mecanismos adicionais para garantir a qualidade e desempenho do serviço final. De fato, existem adaptações para resolver problemas específicos tal como possibilitar transmissão ao vivo, como descrito em (CHOE et al., 2007; GALLO et al., 2009) e analisado em (ANNAPUREDDY et al., 2007). Tais abordagens atuam no modo como os *peers* são selecionados e na sequência de obtenção dos pedaços, otimizando estes processos em função das características da aplicação que consome o conteúdo ou da forma de gerenciamento da rede. A Figura 6 ilustra essa pragmática, contrapondo uma rede usando o sistema BitTorrent de forma tradicional (P2P não estruturado) e um sistema BitTorrent otimizado (P2P estruturado).

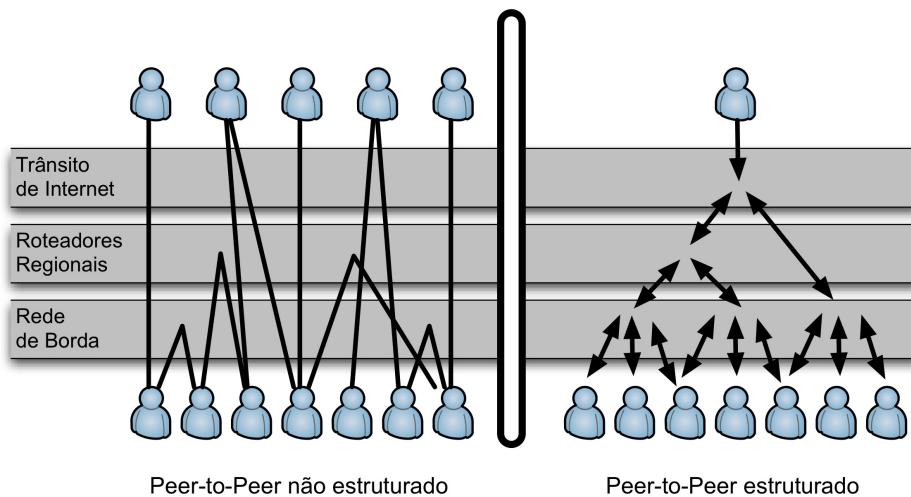


Figura 6: Exemplo de arquiteturas que podem ser empregadas para obter determinadas características de um sistema BitTorrent

A Figura 6 apresenta do lado esquerdo um sistema BitTorrent tradicional que não seleciona os *peers* que são retornados a um *peer* quando ele solicita um conteúdo,

enquanto um sistema estruturado adota critérios pré-definidos para determinar quais e quantos *peers* serão retornados a cada requisição. O objetivo do BitTorrent estruturado na Figura 6 é a diminuição de tráfego na Internet por meio da cooperação entre *peers* locais na ponta da Rede de Borda, o que diminui o tráfego entre provedores de acesso à Internet que estão no nível dos Roteadores Regionais.

Quando se considera a transmissão ao vivo, o sistema deve ser otimizado para que a seleção dos pedaços de um conteúdo seja sequencial a fim possibilitar uma boa QoE. Nesse sentido, estudos como (LI, 2006; HEI et al., 2007; AGARWAL; SINGH; DUBE, 2007) mostram aspectos, arquiteturas e cenários para fazer transmissão ao vivo de vídeo. Arquiteturas desta natureza têm, como objetivo, evitar que os provedores de acesso tenham que contratar enlaces regionais ou Internet com maior largura de banda para fornecerem serviços dentro do *Service Level Agreement* (SLA) contratado pelos seus usuários. No entanto, existe uma grande diversidade de cenários que uma rede P2P pode assumir, bem como formas de organizar os *peers* em níveis para que a cooperação produza os resultados desejados e atenda os requisitos da aplicação considerada. Pode-se afirmar, então, que as arquiteturas de rede empregadas em P2P estão longe de seguir um padrão em decorrência dos requisitos das aplicações que utilizam os seus dados (HAMRA; FELBER, 2005).

2.4 Considerações do Capítulo

Até o ano de 2006, o tráfego P2P figurava como o principal tipo de tráfego em *backbones* da Internet. Contudo esse tipo de serviço não oferece uma interação direta com o usuário visto que o mesmo deve primeiramente obter o conteúdo para depois usá-lo na aplicação fim; por exemplo, o usuário obtém o arquivo de vídeo usando um cliente P2P e somente depois de obter todo conteúdo é que pode visualizá-lo usando uma aplicação de reprodução de vídeos. Ao mesmo tempo, a demanda por conteúdo de vídeos através da Internet está impactando diretamente no volume de tráfego nos

backbones, tomado a primeira posição do P2P em 2006 e assim permanecendo desde então. Tal fato ocorreu principalmente devido à melhora significativa de serviços de banda larga na Rede de Borda e à facilidade de acesso ao conteúdo de serviços de vídeo através do protocolo HTTP (usando navegadores *web*) via CDN. Contudo, essa abordagem segue essencialmente um modelo cliente-servidor que penaliza o sucesso dos provedores, isto é, quanto mais usuários acessando o serviço pior tende a ser a qualidade por ele provida ou torne necessária a contratação de serviços adicionais (e.g., CDN como a Akamai⁵). Uma abordagem possível para resolver este problema é a adoção de sistemas P2P otimizados para atender requisitos de desempenho específicos e já acoplados à aplicação final, permitindo que o usuário final obtenha o vídeo de forma transparente e dispensando etapas adicionais para sua reprodução.

Diferente dos sistemas cliente-servidor, os sistemas P2P têm seu desempenho melhorado com o aumento do número de usuários, com uma possível degradação quando houverem poucos *peers* ou apenas um. Apesar deste potencial, os sistemas P2P existentes em sua maioria não são otimizados para desempenho e preocupam-se essencialmente em manter o conteúdo disponível na rede e incentivar o compartilhamento. Neste contexto, uma das formas mais comuns de otimizar sistemas P2P é o emprego de algoritmos de localidade para otimizar os mecanismos de escolha dos *peers* que serão utilizados para obter um dado conteúdo.

⁵www.akamai.com

3 LOCALIDADE EM REDES P2P

Localidade pode ser definida como a proximidade entre os nós envolvidos em uma determinada operação de rede (DARLAGIANNIS; MAUTHE; STEINMETZ, 2004; MIERS et al., 2009). Esta proximidade pode ser medida usando diferentes critérios, como a localização física dos nós, o número de saltos entre eles, a latência no enlace que os liga, entre outros. Por exemplo, se a proximidade levar em conta apenas a latência do enlace entre os nós de rede, os nós conectados diretamente por uma fibra ótica estarão provavelmente muito “próximos”, mesmo que os mesmos estejam fisicamente localizados em pontos muito distantes um do outro. Um dos objetivos mais importantes dos algoritmos de localidade é o de encontrar um conjunto otimizado de nós capazes de oferecer (dependendo dos critérios adotados para medir proximidade) uma baixa latência de rede, menor tráfego, redução de custos, etc. Algoritmos de localidade podem também ajudar os *peers* a receberem os dados de acordo com certas restrições de tempo; por exemplo, em um sistema de VoD (*Video on Demand*), é importante assegurar a continuidade do vídeo quando o usuário estiver assistindo e obtendo o conteúdo simultaneamente. Neste caso, os algoritmos de localidade podem aumentar a qualidade dos serviços oferecidos por redes P2P, produzindo uma melhor experiência para os usuários. O conceito de localidade pode ser utilizado em qualquer das entidades que compõem uma rede P2P.

3.1 Breve histórico da localidade em redes P2P

Nos primeiros sistemas P2P, os processos de seleção de *peers*, bem como a busca e o gerenciamento dos seus recursos, normalmente não eram otimizados. Com base na análise das estatísticas geradas por esta primeira geração de redes P2P, e também com a maior evolução destes sistemas, ficou claro que as soluções baseadas em P2P eram não somente possíveis, mas também que o desempenho das mesmas podia ser consideravelmente melhorado com o desenvolvimento de novos (e, muitas vezes, simples) mecanismos.

Dentre as diferentes estratégias possíveis, os algoritmos de localidade atraíram uma considerável atenção devido ao seu potencial em fornecer informações úteis sobre o estado da rede, orientando as decisões a serem feitas pelo sistema e, assim, conduzindo a um melhor desempenho e ao uso otimizado dos recursos da rede. De fato, a análise de diversas soluções baseadas em P2P mostra que alguns dos seus problemas poderiam ser prevenidos (ou ao menos atenuados) pelo emprego de técnicas de localidade. Um exemplo é o caso do GNUtella (RIPEANU, 2001), um dos primeiros sistemas P2P a ganhar popularidade, e cuja a estratégia para localizar os conteúdos consiste no envio de requisições por meio de inundação (*flooding*), i.e., o envio é feito a todos os nós da rede de forma indistinta, sem a adoção de qualquer técnica de localidade. Devido à enorme quantidade de tráfego gerada desta maneira, o GNUtella sofre de sérios problemas de escalabilidade (ZHAO; JOSEPH; KUBIATOWICZ, 2002) que poderiam ser mitigados com uma seleção mais cuidadosa de *peers* (ZHANG, 2005). Ao contrário da técnica de *flooding*, estratégias baseadas em servidores podem tirar proveito do estado da rede e das informações de topologia para proporcionar um melhor desempenho. Todavia, tais alternativas podem apresentar problemas como o aparecimento de *hot spots* ou a ocorrência de *flash-crowds* (STADING; MANIATIS; BAKER, 2002; RUBENSTEIN; SAHU, 2005) e, consequentemente, exigem estruturas de rede mais estáveis. Estes inconvenientes motivaram o desenvolvimento de novas soluções para o gerenci-

amento dos *peers*, como a conservação de informações dos *peers* em *Distributed Hash Tables* (DHT) ou o uso de servidores estruturados (XIE et al., 2008; TU et al., 2008). Porém, mesmo sem estes mecanismos, o desenvolvimento de algoritmos de localidade cuidadosamente projetados pode oferecer uma melhor distribuição de carga entre *peers*, reduzindo a probabilidade de eventos indesejáveis como os previamente descritos. De fato, algoritmos de localidade não dependem da solução exata adotada para gerenciar os *peers* ou os recursos em sistemas P2P. Pelo contrário, o uso de localidade pode trazer benefícios a tais mecanismos (por exemplo: a localidade pode ser usada para otimizar a organização dos servidores P2P). Nesse sentido, o BitTorrent é classificado como rede P2P organizada em servidores.

Em um cenário baseado em *trackers* BitTorrent, por exemplo, esses podem utilizar informações de localidade para criar conjuntos otimizados de *peers* sempre que algum conteúdo for solicitado; ao mesmo tempo, os clientes podem usar tais mecanismos para filtrar a lista de *peers* recebida do *tracker*, determinando com quais *peers* devem ser estabelecidas as conexões iniciais. É importante notar, entretanto, que cada aplicação (serviço fornecido, e.g., *Video on Demand* (VoD)) podem adotar tipos diferentes de algoritmos e métricas, de acordo com suas próprias necessidades. A decisão sobre qual é o tipo de localidade mais adequado depende das características da aplicação alvo e também do cenário de rede no qual ela será implementada. Por esta razão, e com o objetivo de desenvolver uma taxonomia de localidade realista, é importante avaliar a forma como estes fatores foram considerados nas propostas de localidade já existentes. Devido à grande importância de se entender como a escalabilidade inerente das redes P2P pode ser combinada com o conceito de localidade, uma comparação coerente e fundamentada entre estas soluções torna-se essencial.

3.2 Métricas de localidade em redes P2P

Métricas¹ são necessárias para avaliar o quanto eficiente é um algoritmo de localidade, e também para prover informações sobre tal algoritmo. Basicamente, métricas são empregadas durante a medição e avaliação de algumas características de um sistema. Desta maneira, elas permitem que se verifique o quanto este sistema satisfaz seus requisitos operacionais. As métricas identificadas para medir localidade, dependendo da estrutura da rede P2P alvo, são:

1. **Proximidade na seleção de nós *overlay*:** definida como a habilidade para explorar, sempre que possível, os recursos disponíveis localmente ao invés de recursos remotos. Esta métrica ajuda a determinar o impacto das técnicas empregadas sobre a escalabilidade do sistema. Aqui, “local” e “remoto” são definidos no contexto de enlaces de rede heterogêneos, com diferenças em termos de latência, capacidade de transmissão ou banda de transmissão, e distância entre os nós de origem (requisitante) e destino. Intuitivamente, esta também é a propriedade que permite a um sistema limitar o impacto de operações locais sobre o desempenho global da rede, tanto durante a execução de operações regulares quanto em situações adversas (ZHAO; JOSEPH; KUBIATOWICZ, 2002).
2. **Penalidade de Atraso Relativo:** é definida como a razão entre as latências observadas ao enviar dados usando uma rede *overlay* e quando os mesmos dados são enviados diretamente (i.e., usando a rede subjacente) (DARLAGIANNIS; MAUTHE; STEINMETZ, 2004); pode, assim, ser medida pela demora adicional de transmissão de pacotes introduzidos pela rede sobreposta quando alguma mensagem é trocada entre dois nós.
3. **Estresse do enlace:** quantifica o uso da rede subjacente pela rede sobreposta. É definido como o número de túneis (canais lógicos) que enviam tráfego sobre um

¹Métricas de software estão relacionadas a indicativos qualitativos e quantitativos.

enlace físico (DARLAGIANNIS; MAUTHE; STEINMETZ, 2004).

4. **Carga adicional de controle:** esta métrica é usada para analisar o desempenho e pode ser avaliada a partir do tempo gasto pelos processos de inserção e obtenção dos nós com os quais um *peer* irá estabelecer comunicação (DARLAGIANNIS; MAUTHE; STEINMETZ, 2004).
5. **Qualidade do caminho de dados:** é utilizada para analisar o desempenho, avaliando a proximidade de rede por meio do “atraso fim a fim”. O atraso fim a fim corresponde ao *Round-Trip Time* (RTT) entre um nó pai e um nó filho dentro da estrutura de uma *overlay* (e.g., árvore *multicast*). (DARLAGIANNIS; MAUTHE; STEINMETZ, 2004).
6. **Contribuição do provedor na aceleração do P2P:** avalia a capacidade do provedor de conteúdo em ajudar a acelerar a distribuição do mesmo (XIE et al., 2008).
7. **Uso eficiente de recursos da rede:** avalia a forma como está sendo realizado o uso dos recursos da rede em comparação com valores de base (*baseline*)², de modo a determinar se e o quanto estes recursos estão sendo bem utilizados. Um sistema pode permitir às aplicações utilizar essas informações sobre o estado da rede, reduzindo, assim, o tráfego no *backbone* da rede e os seus custos operacionais (XIE et al., 2008).

Analizando estas métricas é possível perceber que não existe um conjunto de critérios em comum usado pelos diversos autores da área para compararem suas soluções. No entanto, os diversos algoritmos propostos para aperfeiçoar as diferentes técnicas de localidade utilizam essas métricas de uma forma simples ou híbrida. Neste contexto, os critérios adotados para escolher uma técnica de localidade (ou um conjunto delas) para uma rede P2P dependem do tipo da aplicação alvo.

²Valores Base são pré-definidos de acordo com requisitos de uma aplicação (e.g., tempo de obtenção de um vídeo de 30 minutos) ou segundo critérios próprios do operador da rede.

Nesta tese, para identificar métricas adequadas para algoritmos de localidade, são adotados os conceitos estabelecidos na RFC2330 (PAXSON, 1998) para métricas de rede. Estas métricas de rede podem ser classificadas como básicas e compostas (MOLINA et al., 2006). Uma Métrica Básica (MB) está relacionada a informações simples (por exemplo: o consumo de banda em um enlace), que podem ser obtidas diretamente por meio de uma única medição. Por outro lado, uma Métrica Composta (MC) é obtida através de um conjunto de MBs; por exemplo, o consumo de banda em um enlace fim a fim pode ser calculado usando o consumo de banda dos enlaces que compõem o caminho de dados e, portanto, é considerada uma métrica composta.

Paxson (PAXSON, 1996) também define as métricas como sendo analíticas ou empíricas. Na Métrica Analítica (MA) é usada uma análise matemática (formalismo) dos elementos a serem analisados; as métricas desse tipo são fundamentadas em propriedades teóricas e abstratas do objeto medido. Na Métrica Empírica (ME), diferentemente, é definida diretamente em termos de uma metodologia de medida; por exemplo, elas podem depender de uma implementação específica ou de um cenário de testes. A Tabela 2 apresenta as métricas de localidade previamente listadas, classificando-as de acordo com os critérios definidos por (PAXSON, 1996; PAXSON, 1998).

Tabela 2: Métricas de localidade.

Métrica	MB/MC	MA/ME
Proximidade na seleção de nós <i>overlay</i>	MC	ME
Penalidade de Atraso Relativo	MC	MA
Estresse do enlace	MC	ME
Sobrecarga de controle	MC	MA
Qualidade do caminho de dados	MC	ME
Contribuição do provedor na aceleração do P2P	MC	ME
Uso eficiente de recursos da rede	MC	ME
Latência	MB	MA
Vazão	MB	MA
Caminho de dados mais curto	MB	MA

Conforme observado na Tabela 2, a maioria das métricas são compostas (MC) e empíricas (ME). Embora seja possível comparar os diferentes algoritmos baseando-se somente no tipo de métrica adotada por eles, esta não é uma tarefa trivial.

3.3 Classificação de localidade em redes P2P

Alguns algoritmos de localidade existentes são brevemente descritos nesta seção para exemplificar a sua diversidade, a forma como eles são aplicados em diferentes cenários e o emprego de métricas distintas. No entanto, é possível notar a existência de algumas semelhanças entre diferentes algoritmos com relação ao tipo de classificação adotada (por exemplo: proximidade espacial/geográfica dos *peers*, e aspectos relacionados ao conteúdo). Tal fato indica que diferentes algoritmos de localidade podem ser agrupados de acordo com alguns atributos gerais, permitindo assim uma identificação mais fácil da sua organização e das suas principais funcionalidades. Esta expressiva diversidade leva, assim, à necessidade de usar uma taxonomia que facilite o processo de escolha sobre qual algoritmo adotar em cada cenário de aplicação. A Figura 7 apresenta um modelo de taxonomia para classificação dos algoritmos de localidade, sendo esta taxonomia uma contribuição desta tese.

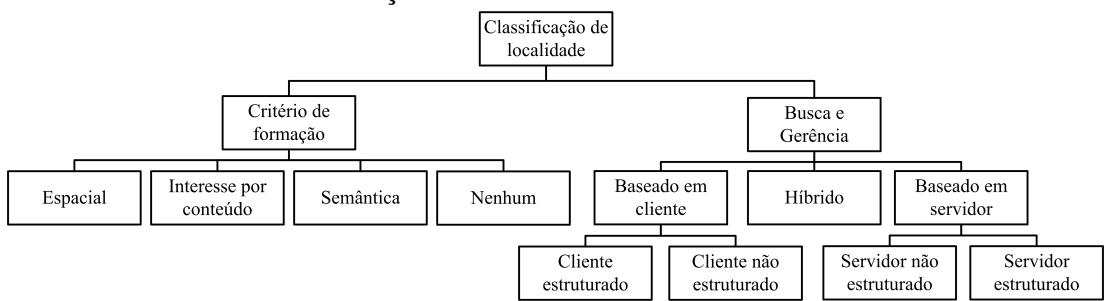


Figura 7: Classificação de localidade para redes P2P

As características consideradas nesta taxonomia são:

- **Critérios de formação:** os critérios adotados pelo algoritmo de localidade para identificar e definir quais *peers* serão usados quando uma operação normal para troca de recursos via P2P é executada. Este item também pode ser definido como

a maneira pela qual os *peers* serão identificados e adicionados a clusters formados empregando-se a informação de localidade, durante o processo de obtenção de conteúdo. As seguintes estratégias podem ser utilizadas:

- **Espacial:** especifica que o escopo da busca por conteúdo e por *peers* é definido com base na informação da posição dos mesmos na estrutura da rede (por exemplo: usando o endereço IP e CIDR) (FREEDMAN; FREUDENTHAL; MAZIERES, 2004; FREEDMAN et al., 2005; YU; LEE; ZHANG, 2005; TU et al., 2005). Esta técnica pode ser vista como uma forma de direcionar os *peers* ao conteúdo (VENUGOPAL; BUYYA; RAMAMOHANARAO, 2006).
- **Interesse por conteúdo:** assume que, se um *peer* *P1* possui algum pedaço de conteúdo no qual outro *peer* *P2* está interessado, é muito provável que *P2* também se interessará por outros dados provenientes de *P1* (SRIPANIDKULCHAI; MAGGS; ZHANG, 2003; SRIPANIDKULCHAI, 2005; VENUGOPAL; BUYYA; RAMAMOHANARAO, 2006; CAI; WANG, 2006).
- **Semântica:** a descoberta de serviços usa como palavras-chave vetores de atributos criados a partir da descrição do serviço, ao invés de identificadores baseados em *hashes* o quê permite encontrar aspectos mais específicos de similaridade (e.g., localização do *peer*) entre serviços ou seus recursos. Esta estratégia permite que os sistemas descubram serviços semanticamente semelhantes por meio de um *hashing* sensível à localidade (YAN; ZHAN, 2004).
- **Nenhum:** baseadas em *flooding*, com o envio de requisições para todos os *peers* indistintamente (ZHAO; JOSEPH; KUBIATOWICZ, 2002).
- **Busca e gerência:** define quais elementos (*peers* e/ou servidores) são responsáveis pelo gerenciamento de clusters de localidade, bem como pela busca por conteúdos/*peers* nestes clusters. As diferentes estratégias utilizadas podem ser classificadas como:

- **Baseada em cliente:** os *peers* são auto-administrados.
 - * **Cliente não estruturado:** os *peers* organizam-se em um mesmo nível. O gerenciamento do sistema e a busca por recursos são executados pelos *peers* de forma distribuída, por meio de uma determinada métrica.
 - * **Cliente estruturado:** os *peers* elegem alguns nós autorizados especiais (por exemplo: super-nós) que irão, com o auxílio de uma determinada métrica de localidade, controlar as tarefas de busca e gerenciamento.
- **Baseada em servidor:** servidores administram os *peers* e auxiliam na busca por recursos, usando métricas pré-definidas de localidade para tal tarefa; adicionalmente, os servidores também podem definir clusters de *peers* usando esta métrica. Dependendo de como os servidores se organizam, a solução pode ser classificada como:
 - * **Servidor não estruturado:** um ou mais servidores recebem requisições dos *peers* por conteúdo e por recursos da rede disponíveis nas suas proximidades, mas os servidores não possuem qualquer organização específica.
 - * **Servidor estruturado:** um ou mais servidores recebem requisições dos *peers* por conteúdo e por recursos disponíveis nas suas proximidades; no nível dos servidores, há alguma forma de organização (por exemplo: hierárquica).
- **Híbrido:** *peers* e servidores interagem; eles podem assumir diferentes níveis de responsabilidade, mas cooperam de alguma maneira para executar as operações de busca e gerenciamento.

3.4 Principais algoritmos de localidade para redes P2P

Alguns algoritmos de localidade existentes são brevemente descritos nesta seção para ilustrar sua diversidade, a forma como eles são aplicados em diferentes cenários e o emprego de métricas distintas. Os principais algoritmos são:

- **Localidade Geográfica dos Prefixos IP:** Este algoritmo usa o endereço *Internet Protocol* (IP) dos *peers* como base para definir áreas de localidade na rede P2P. (FREEDMAN; FREUDENTHAL; MAZIERES, 2004; FREEDMAN et al., 2005).
- **Leopard:** Incorpora, de forma inerente, um sistema de coordenação virtual na rede P2P. Neste sistema, assim que um nó é admitido na rede, ele é atribuído a uma posição no espaço geográfico de nós. Obtém-se, assim, uma relação de vizinhança que reflete a proximidade dos nós com relação a um ponto inicial (YU; LEE; ZHANG, 2005).
- **Anysee:** Este algoritmo propõe a organização da estrutura de rede em diferentes níveis, levando à construção de uma hierarquia de *peers*. A ideia principal neste caso é usar a posição dos clientes finais dentro da hierarquia para construir uma rede *overlay* com suporte a localidade e organizada como uma árvore de distribuição de dados, de tal maneira que usuários próximos na rede subjacente possam se organizar em sub-árvores (LIAO et al., 2006).
- **UCP2P:** Esta solução cria uma estrutura de dados chamada “super objeto”, com a ajuda de um algoritmo de roteamento interno, e tira proveito da localidade espacial(geográfica) de acessos a arquivos. Durante a execução do sistema, os padrões de acesso a arquivos são analisados, e as informações sobre a localização dos arquivos (as quais, provavelmente, serão recuperadas continuamente) são agrupadas em super objetos (XU; HU, 2003).
- **ASPEN:** O projeto da Rede P2P com Adaptatividade Espacial (*Adaptive Spatial*

P2P Network, ou ASPEN) é uma extensão para Redes com Conteúdo Endereçável (*Content Addressable Networks*, ou CAN) para preservar a informação de localidade espacial e, ao mesmo tempo, reter várias das propriedades de balançoamento de carga dos sistemas DHT. Esta solução usa o conceito de “regiões espalhadas”, unidades de distribuição de dados espaciais que otimizam o balançoamento de carga e também a cobertura espacial do processamento de requisições (WANG; ZIMMERMANN; KU, 2005).

- **Atalhos baseados em interesse:** Nesta solução, os *peers* se organizam espontaneamente, formando uma estrutura baseada em interesses sobre uma rede Gnutella já existente (SRIPANIDKULCHAI; MAGGS; ZHANG, 2003). A principal premissa desse algoritmo é a seguinte: se um *peer P1* possuir algum pedaço de conteúdo no qual outro *peer P2* esteja interessado, é muito provável que *P2* também se interessará por outros dados presentes em *P1*. Tomando essa premissa como ponto de partida, pode-se evitar significativamente o *flooding* da rede usando este algoritmo devido a enviar requisições para *peers* específicos ao invés de fazer *flooding*. Adicionalmente, os atalhos são modulares e podem ser usados para melhorar o desempenho de outros mecanismos de localização de conteúdos, o que inclui esquemas de DHT (SRIPANIDKULCHAI; MAGGS; ZHANG, 2003).
- **Localidade Baseada em Semântica:** Em (YAN; ZHAN, 2004), Yan e Zhan propõem uma extensão para a descoberta de serviços, empregando vetores de atributo gerados pela descrição ou conteúdo do serviço, ao invés dos identificadores que usam *hashes* como palavras-chave. Com a utilização de algoritmos de busca baseados em semântica, esta solução pode melhorar a eficácia de mecanismos de descoberta de conteúdo, ajudando o sistema a descobrir serviços semanticamente próximos e também a localizar estes serviços nos nós que fazem parte da rede. A descoberta dos serviços semanticamente próximos é feita através da identificação de atributos em comum nos vetores dos *peers*.

- **Dagstream:** Este arcabouço utiliza algoritmos para organizar os peers em um Grafo Direcional Acíclico (*Directed Acyclic Graph*, ou DAG), resultando em um esquema que apresenta conectividade de rede e resiliência a falhas. Adicionalmente, esta estrutura permite que os *peers* tomem ciência de dados de localidade de uma forma rápida e distribuída, assegurando o uso eficiente dos recursos da rede (LIANG; NAHRSTEDT, 2006).
- **Shark:** Usa uma estratégia híbrida que combina informações de localidade com índices distribuídos para organizar os *peers* e objetos em agrupamentos (*clusters*) semanticamente coerentes. Este esquema foi desenvolvido como uma parte integrante do sistema de gerenciamento de serviços P2P no projeto MMAPPs (*Market Management of Peer-to-Peer Services*) (HOWISON, 2003; MISCHKE; STILLER, 2004; ANNAPUREDDY; FREEDMAN; MAZIERES, 2005).

Aplicando aos algoritmos de localidade aqui descritos a classificação definida na Seção 3.3 chega-se à Tabela 3.

Tabela 3: Classificação dos algoritmos de localidade.

Algoritmo	Critério de formação	Busca e Gerência
GNUTella	Nenhum	Baseado em cliente: não estruturado
Localidade Geográfica dos Prefixos IP	Espacial	Baseado em cliente: não estruturado
Leopard	Espacial	Baseado em cliente: estruturado
Anysee	Espacial	Baseado em cliente: estruturado
UCP2P	Espacial	Baseado em cliente: estruturado
Aspen	Espacial	Baseado em cliente: estruturado
Atalhos baseados em interesse	Interesse por conteúdo	Baseado em cliente: não estruturado
Localidade baseada em semântica	Semântica	Baseado em cliente: não estruturado
Dagstream	Espacial	Híbrido
Shark	Espacial	Híbrido

A análise dos algoritmos de localidade listados nesta seção indica que a localidade espacial é atualmente a técnica mais explorada. Este fato tem duas explicações principais:

- Uma distância maior entre os nós envolvidos em uma comunicação geralmente implica em latências de rede mais elevadas para a transferência de dados; e
- Provedores de rede normalmente pagam por seus enlaces em função do volume de tráfego gerado.

O critério de busca e gerência possui soluções em todos os tipos, não apresentando uma tendência de preferência definida. Este fato ocorre porque cada solução possui objetivos distintos de como organizar os seus recursos para atender aos requisitos das aplicações ou serviços providos. Essa conclusão foi obtida através de uma análise das referências usadas na elaboração da Tabela 3.

3.5 Considerações do Capítulo

As redes P2P, em especial o BitTorrent, possuem características interessantes para distribuição de conteúdo como a interação entre os participantes para obter um conteúdo e não apenas de um ponto central. Entretanto, o controle e critérios para otimizar a distribuição são normalmente sacrificados na busca de manter o conteúdo disponível e incentivar o compartilhamento; sendo isto algo comum em redes P2P públicas. Iniciativas, da comunidade acadêmica e organizações, vem pesquisando formas de controle e otimização da distribuição para redes P2P e nesse contexto os algoritmos de localidade são a abordagem mais comum.

Os algoritmos de localidade possuem métricas específicas, porém que podem ser classificadas por critérios conceituais (Métrica Básica (MB) e Métrica Composta (MC); Métrica Empírica (ME) e Métrica Empírica (ME)). Em geral, as métricas de localidade são MC e ME devido ao fato que os requisitos de aplicações P2P levam em conta a arquitetura da rede P2P para definir os critérios de desempenho mais adequados. Dentre os algoritmos de localidade o principal tipo empregado é localidade espacial.

Os provedores de serviço de vídeo necessitam satisfazer critérios mínimos de desempenho para que os seus serviços sejam populares. Nesse sentido, a principal abordagem para atender essa necessidade é o emprego de algoritmos de localidade.

O conhecimento das principais propostas e soluções de algoritmo de localidade foi muito importante para especificar a solução proposta nessa tese.

4 PROPOSTA DO SISTEMA P2PM

Analisando o protocolo BitTorrent, descrito na seção 2.3, é possível identificar que os algoritmos empregados reforçam o incentivo ao compartilhamento do conteúdo ao mesmo tempo que buscam manter o conteúdo disponível pelo maior período possível. O incentivo ao compartilhamento no BitTorrent é realizado em sua maioria pelo Algoritmo de *Choking*, descrito na seção 2.3.4, que força os *leechers* a compartilharem o conteúdo que estão obtendo para poder continuar *unchoked* com outros *leechers* que estão conectados e fornecendo conteúdo. A manutenção do conteúdo disponível pelo maior tempo possível na rede é uma consequência do Algoritmo Raros Primeiro, descrito na seção 2.3.3, que busca promover o compartilhamento prioritário dos pedaços do conteúdo que possuem menos cópias disponíveis. O Algoritmo Raros Primeiro tem a sua eficiência limitada, porque ele é aplicado pelos *peers* que não possuem uma visão global de toda a rede, mas possuem uma visão apenas dos *peers* que estão se relacionando na troca de pedaços. Além disso, o *tracker* BitTorrent desconhece quais pedaços do conteúdo cada *peer* possui porque ele somente gerencia o interesse dos *peers* por um conteúdo (*swarm*) e não possui qualquer informação sobre o conteúdo disponível em cada *peer*¹.

Apesar do BitTorrent possuir algumas características eficientes e outras limitadas, o protocolo mostra a sua efetividade na distribuição de conteúdos maiores²

¹O conteúdo nos *peers* é gerenciado pelos próprios *peers* usando os algoritmos explicados na seção 2.3.

²Aqui define-se conteúdos maiores aqueles que possuem o tamanho maior que 10Mb, e.g., vídeos em alta definição.

quando comparado com outros protocolos P2P como o FastTrack (usado no Ka-Zaa) (POUWELSE et al., 2004). Uma análise de alguns dos repositórios públicos de torrents gratuitos como o Torrentreactor.net³, ou concentradores de repositórios como o Torrentz⁴, revelam que a principal parte ativa de conteúdo (conteúdo que o *tracker* controla e ainda possui *peers* compartilhando-o) é composta por arquivos com o tamanho mínimo em torno de 170Mb (tendo grande popularidade arquivos com aproximadamente 1200Mb). Estes dados indicam que o BitTorrent é considerado uma boa opção para obtenção de conteúdos maiores.

Compreendendo a forma como o BitTorrent opera é possível identificar que o *tracker* por si só não otimiza a lista de *peers* que retorna a cada novo *peer* que ingressa no *swarm*. Com isso, os pedaços dos conteúdos podem ser obtidos de locais que gerem aumento de tráfego desnecessário nos *backbones* e com desempenho questionável para o *peer*. Essas características não tornam o BitTorrent uma solução atrativa para disponibilização de conteúdo de vídeo comercial (e.g., *Internet Protocol TeleVision* (IPTV)), no qual é normalmente uma exigência ter a capacidade de controle do conteúdo (localização e manutenção) e ser capaz de gerenciar os recursos de rede disponíveis para otimizar o seu uso ao mesmo tempo que mantém a qualidade da prestação do serviço.

O objetivo desta tese é analisar o uso de redes P2P baseadas em BitTorrent para serviços de fornecimento de conteúdo de vídeo com necessidades de desempenho e propor uma solução, aqui denominada *Peer-to-Peer Managed* (P2PM), que leve em consideração: aspectos de localidade, uso dos recursos da rede e políticas de uso desses recursos. Para satisfazer estes aspectos são necessárias mudanças na forma como o BitTorrent realiza algumas de suas operações.

Redes P2P do tipo BitTorrent empregam *trackers* em suas tarefas iniciais⁵ e também para realizar um gerenciamento de alto nível. Na sua operação padrão, este geren-

³www.torrentreactor.net

⁴www.torrentz.eu

⁵Exemplo: inclusão na rede P2P através do ingresso no *swarm* e busca de *peers* interessados em um conteúdo.

ciamento limita-se a organizar os *peers* interessados em um conteúdo (*swarm*), fazer uma identificação básica destes *peers* (definição de IDs) e fornecer listas de *peers* de um *swarm* sempre que um *peer* solicitar o conteúdo relacionado a tal *swarm* (conforme descrito na seção 2.3). Esta lista não sofre nenhum pré-processamento e contém tipicamente até 50 *peers* (valor padrão no BitTorrent, sendo os *peers* escolhidos aleatoriamente) vinculados no *swarm*, posto que o *tracker* não tem informação sobre quais pedaços (partes) do conteúdo cada *peer* no *swarm* possui ou sobre o posicionamento dos *peers* na rede.

Neste capítulo são descritos os principais requisitos a serem atendidos para resolver as questões levantadas (seção 1.2), assim como os pressupostos inerentes ao tipo de cenário em que a solução pode ser aplicada. Uma vez definidos os requisitos é elaborada uma proposta de arquitetura junto com uma descrição dos seus pressupostos e casos de uso. Em seguida são apresentados e discutidos os trabalhos correlatos, buscando identificar os recursos empregados para resolver questões da mesma natureza.

4.1 Especificação de requisitos

Para o desenvolvimento do Sistema P2PM é considerado um ambiente de redes gerenciadas, ou seja, redes nas quais as conexões entre os seus elementos e os próprios elementos podem ser acessados por algum sistema ou solução de gerenciamento (e.g., qualquer Sistema de Gerenciamento baseado no *Simple Network Management Protocol* (SNMP)) para buscar informações sobre as suas condições e configurações. Este cenário possui alguns pré-requisitos:

- Ser capaz de acessar elementos de rede (e.g., roteadores, *switches* de camadas 2 ou 3) a fim de obter informações sobre o seu uso; e
- Ser capaz de identificar e manipular o fluxo de dados que são transmitidos na rede e que são gerados pela solução proposta.

4.1.1 Requisitos funcionais

Como requisitos funcionais do Sistema P2PM foram identificados:

- Oferecer serviços de distribuição de conteúdo de vídeo empregando redes P2P de forma a otimizar o uso dos recursos da rede;
- Possibilitar que o comportamento da distribuição P2P seja configurável e personalizável através de critérios definidos pelo operador do sistema;
- Identificar as condições da rede e usar essa informação para definir a forma como a distribuição P2P deve se comportar;
- Permitir que sejam aplicadas regras de negócios na infraestrutura da rede de acordo com a suas condições;
- Possibilitar a configuração de parâmetros do Sistema P2PM de modo a atender os requisitos de desempenho da aplicação-cliente que usa o conteúdo obtido a partir desse sistema;
- Permitir o ajuste do comportamento da rede gerenciada através de perfis que definem como os recursos são utilizados e, consequentemente, impactam nos custos da rede. Os custos nesse trabalho referem-se tanto ao consumo dos recursos de rede como ao valor financeiro, visto que a contabilização de vários serviços de rede é feita com base na quantidade de dados trafegados em seus enlaces.
- Possibilitar a configuração dinâmica dos recursos controlados pela solução; e
- Minimizar o tráfego P2P nas redes *backbones* e redes regionais.

4.1.2 Requisitos não funcionais

Além dos requisitos funcionais, foram especificados os seguintes requisitos não funcionais para o Sistema P2PM:

- Permitir que um número considerável de usuários acesse os recursos com um desempenho satisfatório⁶, tendo como limite mínimo operacional um total de dois *peers* (visto que a existência de apenas um *peer* equivale a um cenário cliente-servidor);
- Fornecer o conteúdo solicitado de modo a evitar, preferencialmente, a interrupção da reprodução do conteúdo após esta ser iniciada; e
- Fornecer recursos para monitorar as condições do sistema P2PM por parte do operador.

4.2 Descrição da arquitetura proposta

Uma abordagem direta para tratar as questões apresentadas na seção 1.2 é desenvolver um *tracker* com funcionalidades avançadas que permitam gerar listas de *peers*⁷, tendo como base as informações sobre o uso e condição atual da rede (e.g., banda livre no enlace), a topologia da rede e o conteúdo disponível nos *peers* e, como critério ou objetivo, a otimização dos recursos da rede. Estas informações são úteis para criar um grafo ponderado que reflita a topologia da rede e seu estado, assim como as políticas do provedor de rede (ou regras de negócio), conforme detalhado na seção 4.3.1. Deste modo, a lista de *peers* retornada pelo *tracker* não será mais aleatória e sim o resultado de um processamento/cálculo baseado em informações da rede. Entretanto, conforme aumentam o tamanho da rede, o número de conteúdos gerenciados e o número de *peers*

⁶Aqui são considerados como requisitos para uma aplicação de vídeo, para comprovação nesta tese são descritos os requisitos da aplicação escolhida no Capítulo 6.

⁷A lista de *peers* é elaborada pelo *tracker* sempre que um *leecher* entra em um *swarm*.

participando do *swarm*, isso pode implicar em uma possível sobrecarga computacional nesse tipo de *tracker* avançado. Sendo assim, apesar dos benefícios dessa abordagem este *tracker* pode ficar suscetível a problemas de escalabilidade. Com o objetivo de superar esta limitação, e ao mesmo tempo garantir o fornecimento de tal lista otimizada, é proposta nesse trabalho a criação de uma estrutura distribuída de *trackers* hierárquicos. Nesta estrutura, cada *tracker* é responsável por uma determinada parte da rede (chamada aqui de domínio) e a informação é agregada (sintetizada) sucessivamente nos níveis superiores de acordo com sua hierarquia.

Tendo em vista, as principais características do *tracker* proposto são:

- Possui conhecimento sobre a topologia da rede e atualiza a mesma de acordo com a entrada e saída de *peers* (*Churn* (STUTZBACH; REJAIE, 2006));
- Possui informação sobre quais conteúdos (e suas respectivas partes) cada *peer* possui; e
- Atualiza as informações das condições dos enlaces de rede periodicamente usando informações obtidas dos equipamentos de rede (e.g., roteadores e *switches* de camadas 2 e 3) e *peers*.

O resultado destas modificações é um novo projeto fundamentado no protocolo e arquitetura BitTorrent, mas que leva em consideração aspectos relacionados a engenharia de tráfego (REXFORD, 2008) e otimização de redes (WEN; WANG; YANG, 2006). A abordagem de engenharia de tráfego adotada consiste em identificar as condições da rede e usar estas informações coletadas como entrada de um sistema que as avalia regularmente para gerar novas configurações a serem aplicadas na rede de acordo com um conjunto de políticas pré-definidas. Esta abordagem é ilustrada na Figura 8.

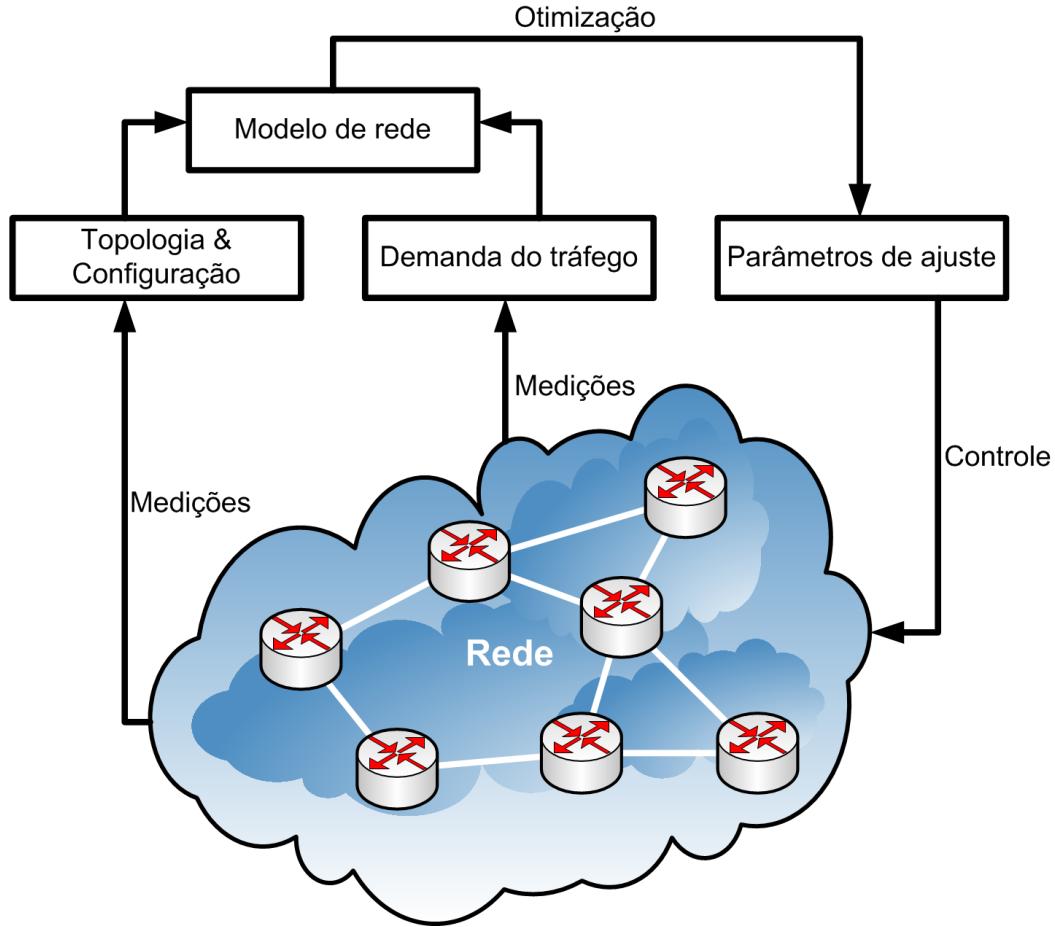


Figura 8: Abordagem para otimizar redes usando engenharia de tráfego de dados.
Adaptado de: (REXFORD, 2006)

Tal abordagem permite que os *trackers* explorem informações sobre a rede (e.g., quando estão selecionando uma lista de *peers*) para modelar o seu desempenho/comportamento, usando para isto um grafo ponderado que representa os elementos da rede e uma fórmula que define os pesos de suas arestas.

4.2.1 Domínio Tracker

O núcleo da arquitetura proposta reside no fato que a rede é partitionada de maneira estática, em escopos chamados de domínios. Para cada domínio é atribuído um único *tracker* que é responsável por todos os elementos de rede e *peers* dentro daquele escopo, criando-se assim vários Domínios *Tracker*. A Figura 9 exemplifica este conceito, mostrando uma arquitetura hierárquica de *trackers* com dois níveis.

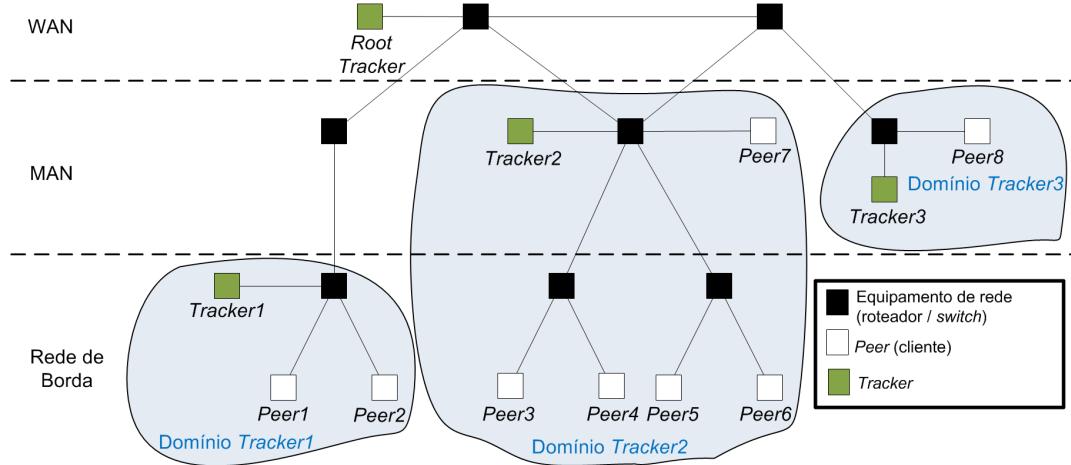


Figura 9: Hierarquia de *trackers*

O exemplo da Figura 9 ilustra uma rede que foi dividida e organizada em três domínios (Domínio *Tracker1*, Domínio *Tracker2* e Domínio *Tracker3*). Pode-se observar que cada domínio possui um *tracker* específico (*Tracker1*, *Tracker2* e *Tracker3*) para gerenciar os *peers* daquele domínio. Além disso, existe no nível superior um Domínio *Root Tracker* que é responsável pelo gerenciamento do escopo que reúne todos os Domínios *Tracker*.

Quando deseja ingressar em um *swarm*, um *peer* sempre conecta-se ao *tracker* que é responsável pelo seu domínio, não importando a disponibilidade daquele conteúdo localmente no seu Domínio *Tracker*. Caso o conteúdo não esteja disponível, o *tracker* do domínio, onde está o *peer* solicitante, reencaminha a solicitação para o próximo *tracker* na hierarquia. Isto ocorre sucessivamente até que a solicitação alcance um *tracker* que tenha o conteúdo disponível em seu domínio ou alcance o *tracker* no topo da hierarquia (chamado de *Root Tracker*), sem que um domínio com o conteúdo seja identificado. Este último caso ocorre apenas quando o conteúdo não existe mais no sistema. Deve-se enfatizar que todos os *peers* notificam o *tracker* responsável pelo seu domínio sobre quais conteúdos e quais partes que eles têm disponíveis. Essas notificações só acontecem entre *trackers* em um nível inferior com o *tracker* no nível imediato superior.

Nesta abordagem, cada domínio é configurado pelo operador de rede em consonância com fatores de agregação⁸, como por exemplo: o número de *peers* a serem administrados em uma determinada região da rede. O esquema hierárquico adotado em conjunto com a especificação adequada dos pesos nos grafos fornece um mecanismo de localidade robusto e objetivo: *peers* próximos serão sempre preferenciais desde que as buscas sejam primeiramente locais; ao mesmo tempo, o sistema suporta uma busca estruturada por *peers* em toda a rede. Cada Domínio *Tracker* pode ser configurado individualmente para empregar os critérios de localidade que melhor se adequam às suas necessidades, conduzindo a uma solução altamente flexível e configurável, além de permitir que o sistema seja personalizado para os mais distintos cenários de rede.

4.2.2 Agregação dos dados

Uma característica importante da solução aqui proposta é que as informações são agregadas à medida que percorrem os níveis dos *trackers* organizados hierarquicamente. Isto é, quanto mais alto for o nível do *tracker*, é mais provável que ele encontre o conteúdo em um domínio mais próximo (segundo os critérios de localidade adotados).

Conforme descrito na seção 4.2.1, cada *tracker* é responsável por uma região da rede (Domínio *Tracker*) e os *peers* desta região irão conectar-se exclusivamente com o *tracker* do seu domínio tanto para obter listas de *peers* de um dado *swarm* como para notificar as partes de cada conteúdo que possui disponível. O *tracker* reporta, para o próximo *tracker* hierarquicamente acima dele, quais conteúdos (e quais partes) estão disponíveis na região que ele gerencia. Sempre que o estado de algum conteúdo é modificado uma mensagem de atualização refletindo a nova disponibilidade de pedaços/conteúdo é enviada para o nível hierárquico superior, que repete o processo até atingir o *Root Tracker*. Desta forma, o *Root Tracker* possui a informação de todos os

⁸Agregação aqui é definida como um conjunto de informações sintetizadas sobre o conteúdo e quais *peers* ou *trackers* possuem o mesmo disponível.

conteúdos e respectivas partes disponíveis em todos os domínios. Isto também permite que os *trackers* próximos ao *Root Tracker* identifiquem todo conteúdo P2P disponível nos níveis inferiores de sua hierarquia, em qualquer momento.

O tráfego de controle resultante destas verificações e atualizações é minimizado pelo fato dos *trackers* apenas se comunicarem com outros *trackers* nesse processo e não estabelecerem conexões com os *peers*. Isto causa um impacto reduzido na escalabilidade do sistema, pois é mais fácil verificar a disponibilidade do conteúdo junto a alguns *trackers* do que por meio de consultas a milhares de *peers*. Ao mesmo tempo, a agregação de dados é alcançada visto que os *trackers* acima na hierarquia sabem apenas quais conteúdos e partes estão disponíveis em um determinado domínio. Neste contexto, sempre que um *tracker* não pode atender a uma solicitação de alguma entidade (seja ela *tracker* ou *peer*), ele irá enviar uma mensagem para o *tracker* imediatamente acima na hierarquia. Este processo é repetido até que a solicitação seja atendida, ou até o *Root Tracker* informar que o conteúdo não está mais disponível na rede.

4.3 Implementação da solução

A solução descrita na seção 4.2 é bastante flexível, podendo conter vários níveis hierárquicos, com um mínimo de dois níveis. A adição de mais níveis somente deve ser realizada quando exigido por questões de escalabilidade, i.e., caso haja necessidade por parte da aplicação usando a rede P2P ou o número de *peers* exceda a capacidade de processamento do *tracker*. Para avaliar a proposta apresentada nessa tese será descrita a implementação de uma arquitetura com a quantidade mínima de *trackers*, isso é: dois níveis hierárquicos. Sendo a nomenclatura empregada para descrever cada hierarquia: *TrackerN* para os *trackers* das extremidades inferiores na hierarquia (folhas) e *Root Tracker* para designar o *tracker* mais alto na hierarquia (raiz).

4.3.1 Modelo do *Tracker*

Para que seja possível elaborar uma lista de *peers* que considere aspectos de localidade e custos, cada *tracker* (inclusive o *Root Tracker*) possui localmente armazenado um grafo ponderado com a topologia da rede do seu Domínio *Tracker*. Este grafo possui a seguinte interpretação:

- Nós (Vértices): representam os *peers* e outros elementos da rede;
- Arestas: representam os enlaces de rede; e
- Peso das arestas: valores computados de acordo com os critérios definidos pelo operador da rede.

Nos grafos ponderados o indicador de distância entre dois *peers* é obtido pelo custo da conexão entre eles. Usualmente o custo é obtido somando todos os pesos que formam o caminho para a conexão entre esses dois *peers*, sendo que o menor valor indica o melhor custo. Portanto, se um *peer* pode obter um conteúdo disponível em diversos outros *peers*, então os *peers* com menor custo representam as melhores escolhas para tal tarefa. Consequentemente, o critério adotado para computar o peso das arestas e o algoritmo para determinar o caminho mais curto implicam diretamente na qualidade da seleção dos *peers* e no impacto do uso dos recursos de rede. Considerando que os enlaces de rede são assimétricos, foi adotado o uso de grafos direcionais para representar os custos tanto em termos de *downlink* como de *uplink*.

A Figura 10 exemplifica esta abordagem, mostrando o grafo direcional relativo ao Domínio *Tracker2* apresentado na Figura 9.

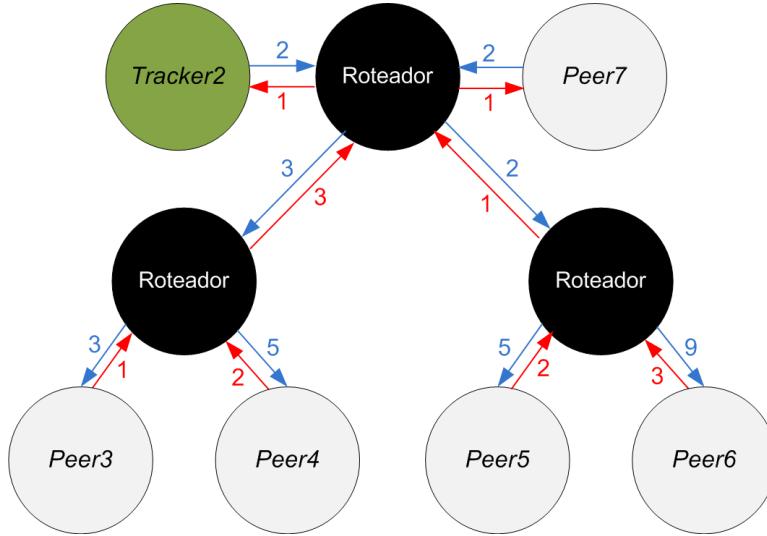


Figura 10: Grafo do Domínio *Tracker2*, enlaces de rede assimétricos

Na Figura 10 podem ser observados os pesos em aresta, sendo as setas vermelhas relacionadas ao *uplink* e as setas azuis ao *downlink*. Este direcionamento é importante para modelar o BitTorrent, visto que o protocolo estabelece tanto conexões de *download* como de *upload* para obter/fornecer conteúdos.

O peso das arestas representa o custo de um enlace e por isso deve ser calculado usando informações atualizadas sobre as condições da rede, as quais devem estar disponíveis no *tracker*. Diversas informações podem ser usadas para este cálculo, incluindo: largura de banda total, largura de banda em uso, latência e taxa de perda de pacotes. Além disso, o operador da rede pode desejar definir valores estáticos para que o peso da aresta reflita regras de negócio (e.g., um valor monetário pago pela quantidade de tráfego transmitido). De uma maneira geral, o grafo do domínio possibilita ao administrador da rede “moldar” o comportamento do tráfego P2P através da arquitetura de *trackers*, fazendo com que a lista de *peers* seja compilada de acordo com condições atuais de carga da rede.

Uma vez que cada *tracker* é responsável pelos *peers* pertencentes a um domínio bem definido, a hierarquia fornece uma localidade inerente: conforme discutido na seção 4.2, se houverem *peers* adequados no Domínio *Tracker* local, estes serão prefe-

ridos se forem a melhor opção após a avaliação do grafo local. Na implementação aqui proposta, o *tracker* foi projetado para retornar uma lista de *peers* contendo o número mínimo de *peers* necessários para obter o conteúdo, de acordo com os critérios de localidade definidos pelo operador. Esta abordagem evita a obtenção de conteúdo de *peers* distantes na rede, que podem implicar o uso desnecessário de recursos de rede. Além disso, os critérios podem ser personalizados para diferentes cenários, permitindo que os proprietários da rede ajustem o comportamento do sistema (influenciado consequentemente no tráfego de rede) de acordo com a demanda. A Figura 11 mostra como um *tracker* trata as solicitações de conteúdo dos *peers* e como a localidade é alcançada.

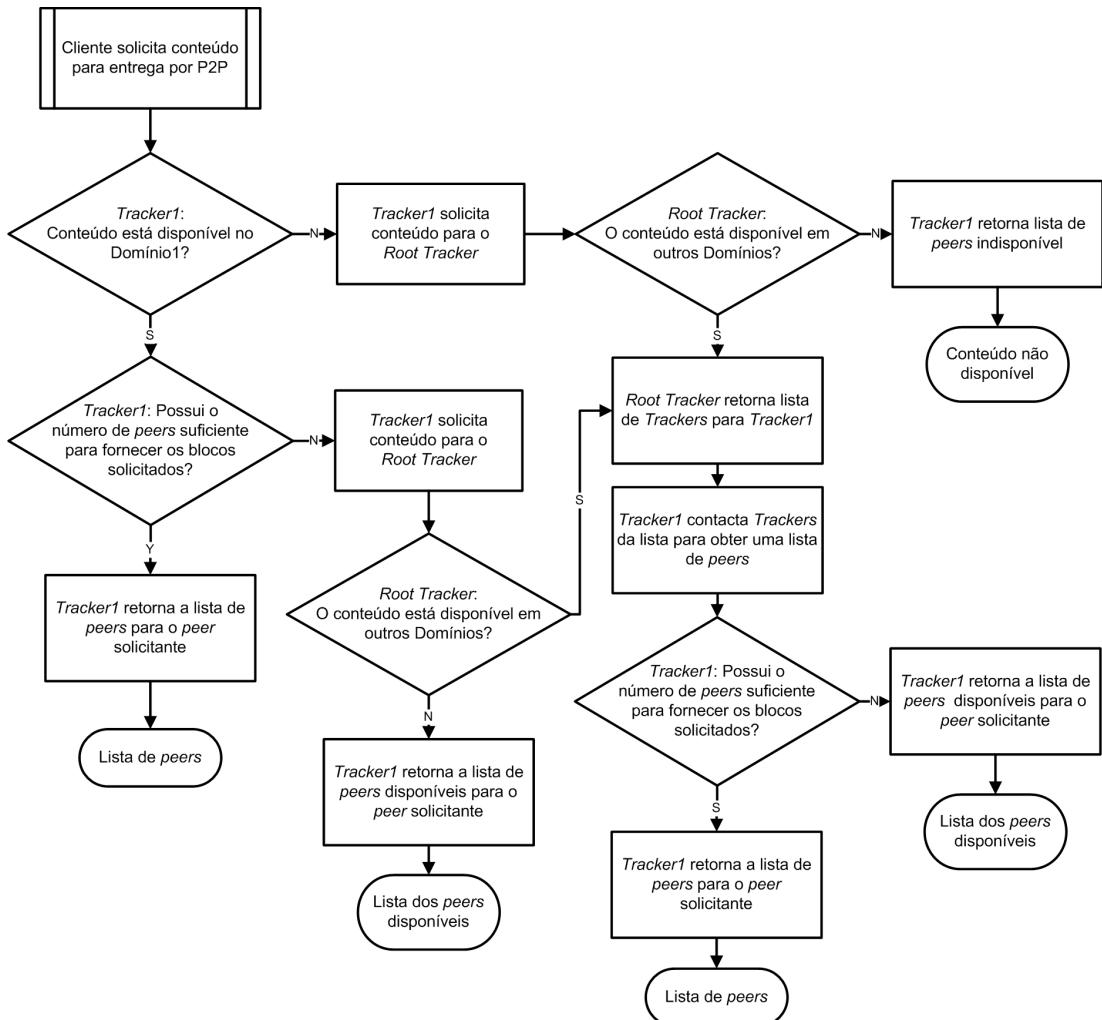


Figura 11: Diagrama de fluxo do tratamento das requisições do *peers* pelo *tracker* empregando localidade

A Figura 11 mostra que toda requisição de conteúdo feita a um *tracker* sempre retornará uma das seguintes respostas:

- **Lista de peers:** Contém uma quantidade suficiente de *peers* para atender a demanda da aplicação no *peer* requisitante. *Peers* adicionais podem ser incluídos como *backup*, tendo em vista questões de *Churn* comuns a redes P2P. Contudo, a inclusão de muitos *peers*, além do minimamente necessário, pode levar a um comportamento indesejado na rede (e.g., existência de tráfego em enlaces que deveriam ser primariamente evitados);
- **Lista de peers disponíveis:** Apesar de serem encontrados *peers* com o conteúdo solicitado, eles podem não ser suficientes para atender a demanda da aplicação no *peer* requisitante. Todavia, em um esforço de atender tal demanda, o *tracker* retorna uma lista com todos os *peers* que encontrou. Esta situação pode implicar na redução do desempenho ou até mesmo impactar na qualidade do serviço da aplicação que está consumindo o conteúdo no *peer* requisitante; e
- **Conteúdo não disponível:** Acontece apenas no caso em que o conteúdo não se encontra na rede, tendo em vista que o operador da rede possui autonomia para incluir e remover conteúdos quando achar necessário. Esta situação deve acontecer raramente, pois está associada principalmente ao caso em que um *peer* obtém os metadados (arquivo .torrent) de um conteúdo pouco antes dele ser removido da rede; ao final do processo de exclusão, o próprio arquivo .torrent também é removido dos *trackers*.

Na arquitetura de dois níveis, apresentada na Figura 9, quando um *peer* P_{req} solicita algum conteúdo que não está disponível no domínio do seu T_{req} , T_{req} irá contactar o *Root Tracker* pedindo uma lista de *trackers* que possuam o conteúdo solicitado. O *Root Tracker* verifica sua tabela para identificar quais *trackers* possuem o conteúdo solicitado presente no seu domínio.

Caso o conteúdo esteja disponível em mais de um *tracker*, o *Root Tracker* usa o seu grafo para selecionar o *tracker* mais apropriado (T_{prov}) para esse conteúdo em função da localização/custo de T_{req} . Então, T_{req} comunica-se com T_{prov} para obter uma lista de *peers* que é adicionada temporariamente ao *swarm* do conteúdo em T_{req} . Estes *peers* temporários são removidos do *swarm* em T_{req} , assim que um *peer* no domínio de T_{req} obtém o conteúdo. Entretanto, caso hajam *peers* P_{req} buscando obter o conteúdo, pode ser necessário manter tais *peers* temporários até que haja *peers* locais em número suficiente no domínio T_{req} para atender à demanda existente.

4.3.2 Computação do peso das arestas do grafo

Existem várias abordagens que podem ser empregadas para definir o peso das arestas de um grafo. O valor do peso pode ser tanto: um valor simples ou o resultado de um cálculo; estático ou dinâmico; e positivo ou negativo. Isso tudo de acordo com o comportamento desejado para o sistema ou até mesmo em função de suas limitações (BALAKRISHNAN, 1997). Valores estáticos do peso são mais simples de se empregarem porque eles não necessitam ser atualizados constantemente; por outro lado, é muito difícil encontrar valores que forneçam bom desempenho e que ao mesmo tempo sejam adequados aos requisitos do sistema e serviços prestados. Por outro lado, valores dinâmicos são capazes de lidar com as mudanças das condições da rede mas dependem de onde e de como a computação dos pesos é executada. Valores de peso negativos ou positivos são uma consequência das fórmulas/critérios adotados. Além disso, alguns algoritmos de menor caminho podem apresentar limitações que devem ser levadas em consideração, como é o caso do Algoritmo de Dijkstra (DIJKSTRA, 1959) que funciona apenas com pesos não negativos. Percebe-se então, que o cálculo do peso da aresta de um grafo deve levar em conta o tipo de valor esperado do peso (negativo / positivo) e as mudanças de peso (estático / dinâmico) para produzir o comportamento desejado do grafo.

Conforme mencionado, o valor do peso das arestas na arquitetura aqui proposta é atualizado periodicamente (dinâmico). Para o cálculo de tais pesos são usados como critérios: um valor de prioridade⁹ e dados obtidos de medições das condições da rede. O valor da prioridade é estático, porém modificável, sendo definido de acordo com as políticas do proprietário da rede (e.g., regras de negócio) e devendo sempre ser empregado como critério. Esse valor é definido como um inteiro não negativo, sendo um a maior prioridade. Pode ser definido um valor de prioridade específico para cada aresta, sendo adotado um valor padrão nas arestas onde um valor específico não foi definido. Esses valores de prioridade (específicos e padrão) são definidos no perfil de cada grafo (*tracker* ou *Root Tracker*). Os valores oriundos das medições de rede podem ser escolhidos de acordo com a disponibilidade a fim de que também sejam usados no cálculo do peso da aresta, por exemplo: dados existentes em uma Base de Dados Gerenciamento (*Management Information Base (MIB)*) obtidos por requisições SNMP. A capacidade total do enlace e o uso atual da banda do enlace são critérios bastante comuns, sendo também valores não negativos. Uma flexibilidade extra oferecida é o fato dos valores a serem medidos também poderem ser definidos individualmente para cada enlace, sendo estes critérios também armazenados no perfil de cada grafo.

Apesar da simplicidade dos valores usados como entrada para o cálculo do peso de cada aresta, a elaboração da fórmula de cálculo é um fator crítico para se obter o comportamento e o nível de desempenho desejados na rede. Uma formulação errada pode conduzir a problemas que vão desde o comportamento indevido do sistema até o baixo desempenho dos serviços disponíveis.

O sistema aqui usado para exemplificar a proposta de arquitetura segue as seguintes premissas para otimizar o cálculo dos pesos e simplificar a identificação do menor caminho entre *peers*:

⁹Aqui o valor prioridade representa uma informação das regras negócios que é incluída no cálculo da aresta. Outra informação que pode ser incluída é valor do Mb trafegado no enlace, por exemplo.

- O peso da aresta é sempre um valor não negativo;
- O peso da aresta deve ser atualizado periodicamente segundo um intervalo previamente definido no perfil de cada grafo; e
- Os valores obtidos nas medições da condição da rede devem estar disponíveis para consulta nos domínios em que forem usados, sendo os mesmos aferidos pelo *tracker* responsável pelo respectivo domínio.

Uma observação importante deve ser feita sobre os valores obtidos de medições da rede: a atualidade dos mesmos é crucial para o correto funcionamento do sistema, pois valores desatualizados podem conduzir o sistema a escolhas de caminho indevidas e por consequência a reações indesejadas. Nesse sentido, existe um desafio em identificar o intervalo entre as medições capaz de equilibrar a carga na rede com tráfego de controle e a necessidade de manter o grafo suficientemente atualizado para fazer frente a mudanças nas condições da rede. Aqui cabe ao operador da rede avaliar os melhores intervalos de acordo com a capacidade da rede e desejo de desempenho do sistema.

4.3.2.1 Algoritmo de menor caminho

Independentemente do cálculo dos pesos das arestas, o algoritmo de menor caminho (*Shortest Path Algorithm* (SPA)) é aplicado para descobrir o melhor caminho entre dois *peers* com base em um grafo de rede. Existem vários algoritmos propostos para solucionar o problema do menor caminho, sendo que a maioria deles consiste em variações dos seguintes algoritmos:

- Algoritmo de Dijkstra (DIJKSTRA, 1959);
- Algoritmo de Bellman-Ford (BELLMAN, 1958);
- Algoritmo de Floyd-Warshall (FLOYD, 1962); e
- Algoritmo de Johnson (JOHNSON, 1977).

Por exemplo, os algoritmos descritos em (AHUJA et al., 1990; WEN; WANG; YANG, 2006; VASSILEVSKA, 2008) são essencialmente baseados no Algoritmo de Dijkstra para encontrar os menores caminhos e no Algoritmo de Bellman-Ford para tratar de pesos negativos. Outras propostas também foram desenvolvidas para solucionar problemas específicos como grafos de tamanho e composição específicos((GALIL; MARGALIT, 1997; SHAIKH-HUSIN; HANI; SENG, 2002)) e não são abordadas aqui por não estarem relacionadas ao foco desta tese.

O pressuposto de valores não negativos possibilita a adoção do Algoritmo de Dijkstra, que foi o escolhido para ser usado nesta tese por sua simplicidade e desempenho, características importantes em sistemas nos quais se busca escalabilidade. Contudo, e apesar do Algoritmo de Dijkstra encontrar o melhor caminho, os protocolos de rede padrão (roteamento estático, *Open Shortest Path First* (OSPF), *Intermediate System to Intermediate System* (IS-IS), etc.) podem selecionar caminhos diferentes para os *peers* obterem os conteúdos. Com o objetivo de assegurar que o caminho escolhido no grafo de rede seja o mesmo utilizado para a comunicação de rede, é adotado o uso do recurso de *Source Routing* definido pela RFC1702 (HANKS et al., 1994). Basicamente, isto possibilita pré-definir no cabeçalho IP (*Internet Protocol*) os saltos que um pacote deve realizar para alcançar o destino, evitando que os roteadores apliquem seus procedimentos padrões de roteamento. O uso de *Source Routing* implica na modificação do protocolo BitTorrent para que, juntamente com a lista de *peers* retornada pelo *tracker*, seja também fornecido o caminho que deve ser empregado na comunicação com cada *peer* da lista. Contudo, caso haja apenas um caminho possível entre os *peers*, o uso de *Source Routing* é desnecessário visto que só há um caminho (rota) possível.

4.3.3 Hierarquia de *trackers* e agregação dos dados

Os grafos dos *trackers* (rede relativa ao Domínio *Tracker*) são empregados para calcular o melhor caminho (rota) entre a origem (*peer* solicitando um conteúdo) e o destino (*peers* que possuem o conteúdo solicitado). Entretanto, entre todos os *peers* disponíveis, é possível que apenas um subconjunto deles tenha as partes (pedaços) do conteúdo que são de interesse do *peer* requisitante. Para prevenir a seleção de *peers* inúteis (fazem parte do *swarm*, mas não possuem as partes no qual o solicitante está interessado), o *tracker* possui localmente a sua própria tabela (Tabela de Conteúdo) contendo as informações dos *peers* que estão ativos e os conteúdos (e suas partes) neles disponíveis. O *Root Tracker* também possui uma tabela similar que relata informação similar, mas relacionada a *trackers* ao invés de *peers*. Com base nessas informações, quando um *peer* solicita um conteúdo, deve ser identificado um subconjunto de *peers* que contenha somente os *peers* que possuem o conteúdo solicitado; esta lista será usada como entrada para o algoritmo de menor caminho. Especificamente, o *tracker* verifica se o conteúdo (partes solicitadas) está disponível da seguinte maneira:

1. O *tracker* verifica sua própria Tabela de Conteúdo (TC) buscando identificar se existem *peers* com o conteúdo solicitado. Caso o conjunto de *peers* obtidos dessa forma seja suficiente para satisfazer o *peer* solicitante, o *tracker* retorna essa lista;
2. Caso o primeiro passo não seja suficiente, o *tracker* tenta localizar se o conteúdo está disponível na rede através de encaminhamento de requisição ao nível hierárquico superior sucessivamente até atingir o *Root Tracker*.

Ambos os passos necessitam que o *tracker* ou *Root Tracker* verifique a disponibilidade do conteúdo no seu domínio. Esta verificação consiste em consultar o ID do Conteúdo (como acontece no Protocolo BitTorrent) na TC mantida pelo *tracker*, que é atualizada de acordo com as informações fornecidas pelos *peers* dentro do Domínio

Tracker. Os campos essenciais para implementação da TC dos *trackers* são:

- **ID do Conteúdo:** identificador único do conteúdo, válido para todo o sistema;
- **ID do Peer:** identificador único do *peer*; e
- **Blocos de Conteúdo:** mapa de bits contendo todos os blocos (partes) disponíveis do conteúdo no *peer*.

A utilização das Tabela de Conteúdo (TC) é outra importante modificação no protocolo BitTorrent que é proposta nesta tese. Isto é necessário porque no protocolo BitTorrent padrão o *tracker* não tem conhecimento de quais partes do conteúdo estão disponíveis em cada *peer*, mas apenas que ele tem interesse no conteúdo; a exceção é o caso dos *seeders*, que por definição tem todas as partes do conteúdo. A inclusão de uma nova entrada na TC ocorre sempre que um *peer* ingressa em um *swarm*: Adicionalmente, a informação dos registros da tabela é atualizada com as informações fornecidas pelos *peers*, que notificam o *tracker* do seu domínio sempre que obtém um pedaço do conteúdo. Analogamente, um registro é excluído da tabela quando o *peer* elimina o conteúdo localmente, bem como quando o *peer* se desconecta do *swarm* ou ultrapassa seu tempo limite (*timeout*) de conexão. Os *peers* enviam periodicamente mensagens para o *tracker* do seu domínio informando que estão ativos (e.g., mensagens do tipo *keep alive*); somente se o *tracker* não receber nenhuma mensagem do *peer* (quer seja de ativo ou novas partes disponíveis) em um intervalo de tempo limite é que este será desconectado e seus registros relacionados apagados da Tabela de Conteúdo.

A Tabela de Conteúdo do *Root Tracker* é mantida por ele mesmo e atualizada conforme as informações fornecidas pelos níveis inferiores da hierarquia. Aqui percebe-se que quando um novo conteúdo é obtido por um *peer*, esta informação é propagada hierarquicamente até chegar ao *Root Tracker*. Contudo, cada *tracker* verifica se é necessário propagar a informação para níveis superiores, a fim de evitar tráfego de controle adicional causado por uma informação redundante. Essa verificação é feita analisando

se o conteúdo relatado pelo *peer* já estava disponível em outro *peer* do domínio, caso não esteja é necessário propagar a informação.

A Tabela de Conteúdo do *Root Tracker* é composta pelos seguintes campos:

- **ID do Conteúdo:** identificador único do conteúdo, válido para todo o sistema;
- **ID do Tracker:** identificador único do *tracker* no sistema; e
- **Blocos de Conteúdo:** um mapa de bits contendo todos os blocos/partes disponíveis do conteúdo nos níveis inferiores do sistema, i.e., de todo o sistema.

É definido que quando um *tracker* T começa a gerenciar um novo conteúdo, um novo registro necessita ser criado na Tabela de Conteúdo (TC) do *Root Tracker*. Consequentemente, também fica especificado que quando é realizada uma atualização de uma entrada da TC de T , deverá ser verificada a necessidade de propagar a atualização para os níveis superiores. Assim, sempre que um *peer* termina de obter uma nova parte do conteúdo esse processo de atualização deve ser executado. Por fim, um registro da TC do *Root Tracker* somente é excluído quando o conteúdo não estiver mais disponível em nenhum *tracker* da rede. A Figura 12 exemplifica o Domínio *Root Tracker* com base nos domínios apresentados Figura 9.

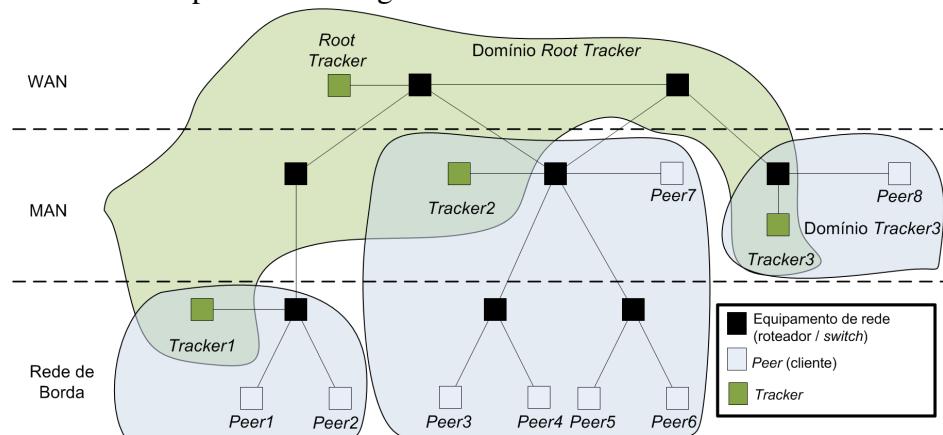


Figura 12: Domínios dos *trackers* e *Root Tracker*

As Tabelas 4, 5 e 6 exemplificam Tabela de Conteúdo (TC) do *Tracker1*, *Tracker2* e *Tracker3* respectivamente. Novamente, deve ser observado que cada *tracker* possui conhecimento apenas dos conteúdos e respectivas partes que estão no seu domínio.

Tabela 4: Exemplo de Tabela de Conteúdo dos *trackers* para o domínio do *Tracker1*

ID do Conteúdo	ID do Peer	Blocos de Conteúdo
100001	Peer1	0000111000
100001	Peer2	1100011111
100002	Peer1	1111100000

Tabela 5: Exemplo de Tabela de Conteúdo dos *trackers* para o domínio do *Tracker2*

ID do Conteúdo	ID do Peer	Blocos de Conteúdo
100001	Peer3	0000111100
100001	Peer5	1111000000
100002	Peer4	1111100000
100003	Peer6	0001111000

Tabela 6: Exemplo de Tabela de Conteúdo dos *trackers* para o domínio do *Tracker3*

ID do Conteúdo	ID do Peer	Blocos de Conteúdo
100001	Peer8	1111111111
100002	Peer8	1111111111
100003	Peer8	1111111111

Analizando a Tabela 4 pode ser observado que nenhum *peer* possui o conteúdo 100001 completo. O *Tracker1*, através da agregação dos dados, consegue identificar com uma simples operação *OR* (OU lógico) que este conteúdo não está disponível por completo em seu domínio.

Diferente do que ocorre no protocolo BitTorrent original, onde o *tracker* é um mero fornecedor de lista de *peers*, na solução proposta existe uma comunicação frequente entre os *peers* e o *tracker* para prover informações sobre o conteúdo e as condições da rede do cliente que podem ser usadas na atualização do grafo de rede. Desta forma, é possível manter as informações sobre as condições da rede e sistema atualizadas; contribuindo para uma melhora no grau de eficiência do sistema como um todo através de informações precisas da relação conteúdo *x peer*. Por este motivo o sistema foi especificado para que sempre que um registro for inserido, modificado ou excluído nas tabelas (TC) o *tracker* envie notificações para os níveis superiores através de informações compiladas. Essa compilação é computada através da aplicação de uma operação lógica *OR* sobre o campo Blocos de Conteúdo dos registros que possuam o mesmo ID de Conteúdo.

A Tabela 7 exemplifica a Tabela de Conteúdo (TC) do *Root Tracker* resultante da agregação dos dados de acordo com as informações recebidas do *Tracker1*, *Tracker2* e *Tracker3*.

Tabela 7: Exemplo de Tabela de Conteúdo do *Root Tracker*

ID do Conteúdo	ID do <i>Tracker</i>	Blocos de Conteúdo
100001	<i>Tracker1</i>	1100111111
100001	<i>Tracker2</i>	1111111100
100001	<i>Tracker3</i>	1111111111
100002	<i>Tracker1</i>	1111100000
100002	<i>Tracker2</i>	1111100000
100002	<i>Tracker3</i>	1111111111
100003	<i>Tracker2</i>	0001111000
100003	<i>Tracker3</i>	1111111111

Na Tabela 7 pode ser percebido que o *Root Tracker* não possui nenhuma informação sobre os *peers*, visto que essas não são necessárias para o funcionamento do serviço de localidade no nível do *Root Tracker*. Portanto, o *Root Tracker* concentra informações sobre o conteúdo disponível em todos os *trackers* no domínio do operador, usando o ID do Conteúdo para indexar esta informação. Caso um conteúdo esteja disponível em múltiplos domínios e não esteja disponível no domínio do *peer* requisitante, então *Root Tracker* escolherá o domínio mais próximo (segundo o grafo do *Root Tracker*) do domínio onde está locado o *peer* requisitante.

A adoção das TC para os *trackers* das extremidades (folhas do grafo) permite ao operador da rede identificar como o conteúdo está distribuído na rede. Portanto, sempre que um *peer* solicita uma parte de um conteúdo, o *tracker* é capaz de encontrar todos *peers* com essas partes de maneira otimizada e faz isso independente dos *peers* com o conteúdo estarem ou não no seu domínio.

4.3.4 Manutenção dos grafos

Conforme descrito neste capítulo, o serviço de localidade P2P é obtido através do uso de uma estrutura de *trackers* hierárquicos que otimizam a lista de *peers* que é retornada sempre que algum conteúdo é solicitado. Esta otimização é realizada usando um grafo ponderado que possui seus parâmetros definidos de acordo com um arquivo de configuração chamado perfil do grafo, onde estão contidas a topologia básica da rede e os critérios e fórmulas para calcular o peso das arestas.

É importante ressaltar que várias das informações contidas no perfil do grafo são personalizáveis, o que significa que o sistema pode ser configurado para atender diferentes requisitos de desempenho. O operador da rede pode ter a necessidade de alterar a topologia da rede ou até mesmo o seu comportamento. Nesse caso, é necessário o gerenciamento dos arquivos de perfis dos grafos assim como orquestrar a sua substituição. Com este objetivo foi adicionado um serviço de gerenciamento de perfis, que é provido pelo Módulo Gerenciador de Perfis (MGP).

O MGP nada mais é que um serviço que possui um conjunto de perfis para os *trackers* e *Root Tracker* e as condições que indicam a ativação de cada perfil em um dado domínio. Para permitir a substituição dos perfis nos *trackers* e *Root Tracker*, estes foram alterados para ter a capacidade de verificar tais condições. O MGP realiza essa verificação através da monitoração constante das condições que podem levar a uma substituição de perfil em um *tracker* ou até mesmo em todo o sistema. Um exemplo é uma condição de dia e hora para substituir o grafo por outro no qual os critérios ou fórmulas sejam mais adequados para a demanda dos clientes naquele horário ((SANDVINE, 2012) apresenta dados que mostram como o volume de tráfego na rede varia de acordo com o período do dia).

A informação da topologia no perfil de grafo contém apenas os elementos estáticos controlados pelo operador da rede porque não inclui os *peers*, os *peers* são variáveis em

disponibilidade (*Churn*), número e posicionamento na topologia. Isto é: a topologia da rede reflete apenas a disposição dos elementos controlados pelo operador da rede (e.g., enlaces, *switches*, *caches* e roteadores). Além disso, para cada critério usado é definida a forma de obter os dados que são necessários; e.g., no caso de uma informação disponível na MIB (SNMP) de um roteador, deve-se fornecer o endereço na MIB do objeto que será usado nos cálculos. Consequentemente, os *trackers* necessitam empregar alguns procedimentos sobre os perfis de grafo para inserir informações (e.g., *peers* e valores) e mantê-las atualizadas. A Figura 13 mostra a representação da máquina de estados do *tracker* para a manutenção do seu grafo.

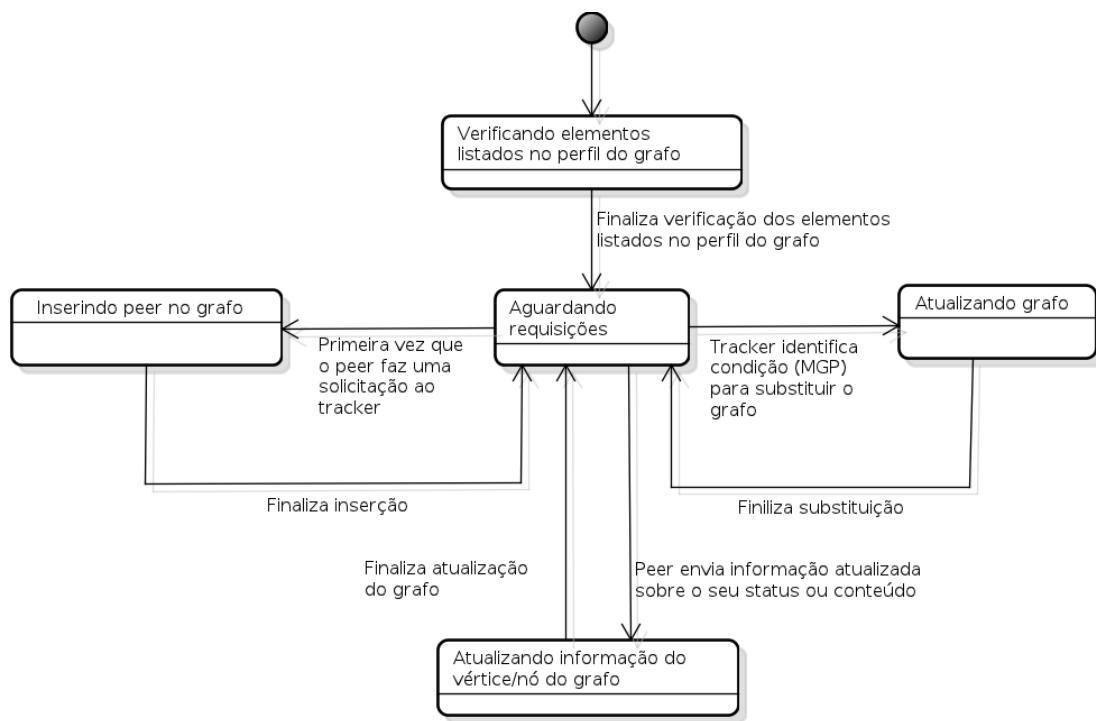


Figura 13: Máquina de estados da manutenção do grafo nos *trackers*

Para compreender melhor os estados da máquina de estados da Figura 13, tem-se:

- **Estado 1 - Verificação elementos listados no perfil do grafo.** O *tracker* primeiro verifica a existência de todos os elementos listados no grafo, ao mesmo tempo que solicita os dados para poder calcular os pesos das arestas pela primeira vez. Esta verificação é necessária para identificar a existência de erros no perfil do grafo ou a indisponibilidade de algum elemento listado.

- **Estado 2 - Inserção do *peer* no grafo.** A inserção de um *peer* no grafo é realizada quando esse estabelece conexão, pela primeira vez, com o *tracker*. Neste momento o *tracker* recebe do *peer* as seguintes informações: endereço IP/Máscara, informações da condição de rede, ID do *Peer*, IDs de Conteúdo (disponíveis no *peer*) e seus respectivos Blocos de Conteúdo. O *tracker* irá usar o endereço IP/Máscara para identificar em qual elemento de rede (vértice/nó do grafo) o *peer* deve ser conectado. Depois do *peer* ser inserido no grafo, o *tracker* irá atualizar a sua TC usando as informações fornecidas pelo *peer*.
- **Estado 3 - Atualização do vértice/nó do grafo.** As atualizações dos dados usados dos elementos de rede é feita periodicamente segundo um intervalo de tempo definido no perfil do grafo. Assim, cabe ao *tracker* fazer as requisições aos elementos de rede. Toda vez que um *peer* finaliza a aquisição de um novo conteúdo ou bloco, ele informa o *tracker*, anunciando a sua disponibilidade como *seeder*. Os *peers* são removidos do grafo sempre que acontece um evento de saída, i.e., quando o *peer* deixa o sistema espontaneamente ou devido a estouro do tempo limite da ausência de mensagens do *peer*.
- **Estado 4 - Atualização do grafo.** Pode ocorrer de o *tracker* já ter o seu grafo totalmente atualizado e com todos os *peers* cadastrados, mas uma determinada condição (definida no MGP, como por exemplo um horário) pode exigir a aplicação de um outro perfil de grafo no *tracker*. Nesta situação, o *tracker* armazena os dados e informações dos *peers* do grafo anterior enquanto verifica os elementos listados no perfil que será aplicado. Depois disso, o *tracker* irá incluir todos os *peers* armazenados no novo grafo de acordo com os seus endereços IP.
- **Estado 5 - Aguardando requisições.** Estado padrão que o *tracker* permanece quando não há nenhuma requisição ou logo após ser inicializado.

O *Root Tracker*¹⁰ realiza operações similares, porém um pouco mais simplificadas por não necessitar lidar com *peers*. Os *trackers* possuem uma disponibilidade bem mais estável do que os *peers* e as inclusões/exclusões de *trackers* não são operações frequentes, sendo realizadas apenas para fins de manutenção. Ademais, todos os elementos de rede dentro do Domínio *Root Tracker* já estão listados no seu arquivo de perfil e desse modo não é necessária a inclusão/remoção de vértices. A Figura 14 mostra a representação da máquina de estados do *Root Tracker* para a manutenção do seu grafo.

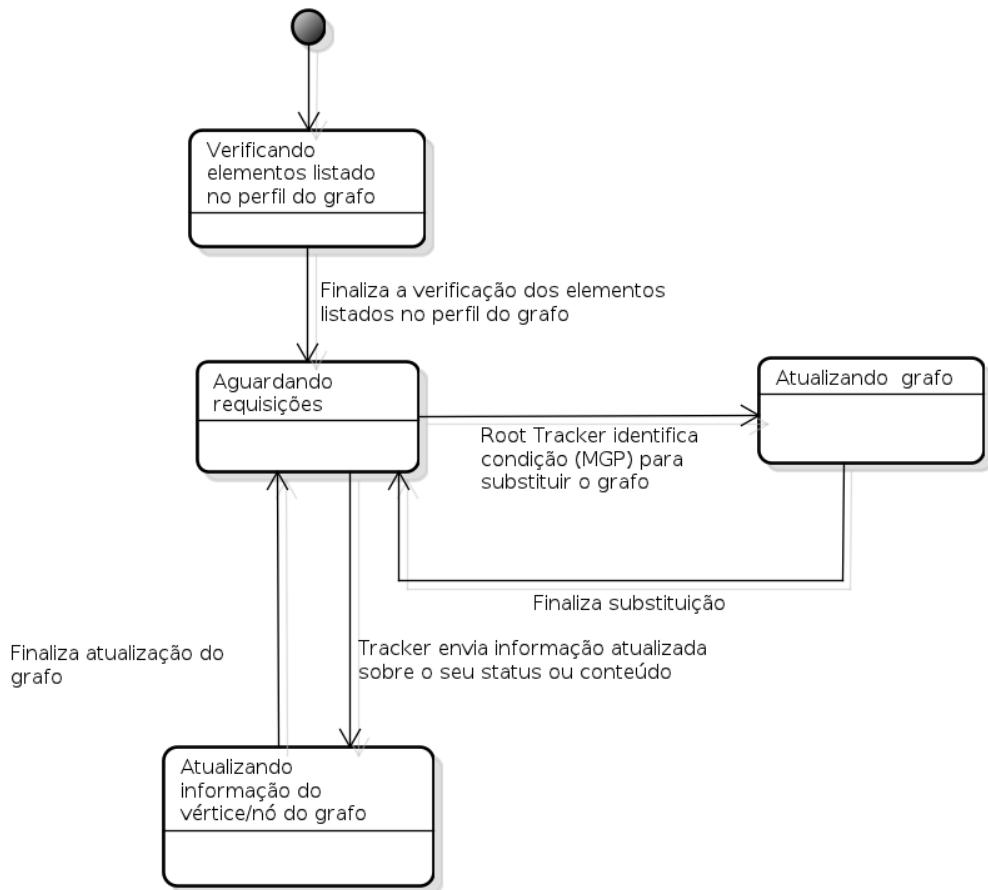


Figura 14: Máquina de estados da manutenção do grafo no *Root Tracker*

A Figura 14 permite visualizar que inicialmente é feita uma verificação de todos os elementos listados no perfil, do mesmo modo que ocorre nos *trackers*. Terminada a verificação, o *Root Tracker* possui o grafo atualizado e estará apto para atualizar suas informações com base nos dados fornecidas pelos *trackers* e elementos de rede.

¹⁰*Trackers* intermediários também se encaixam nessa situação.

A Figura 15 mostra o fluxograma do processo de verificação dos elementos no *Root Tracker*.

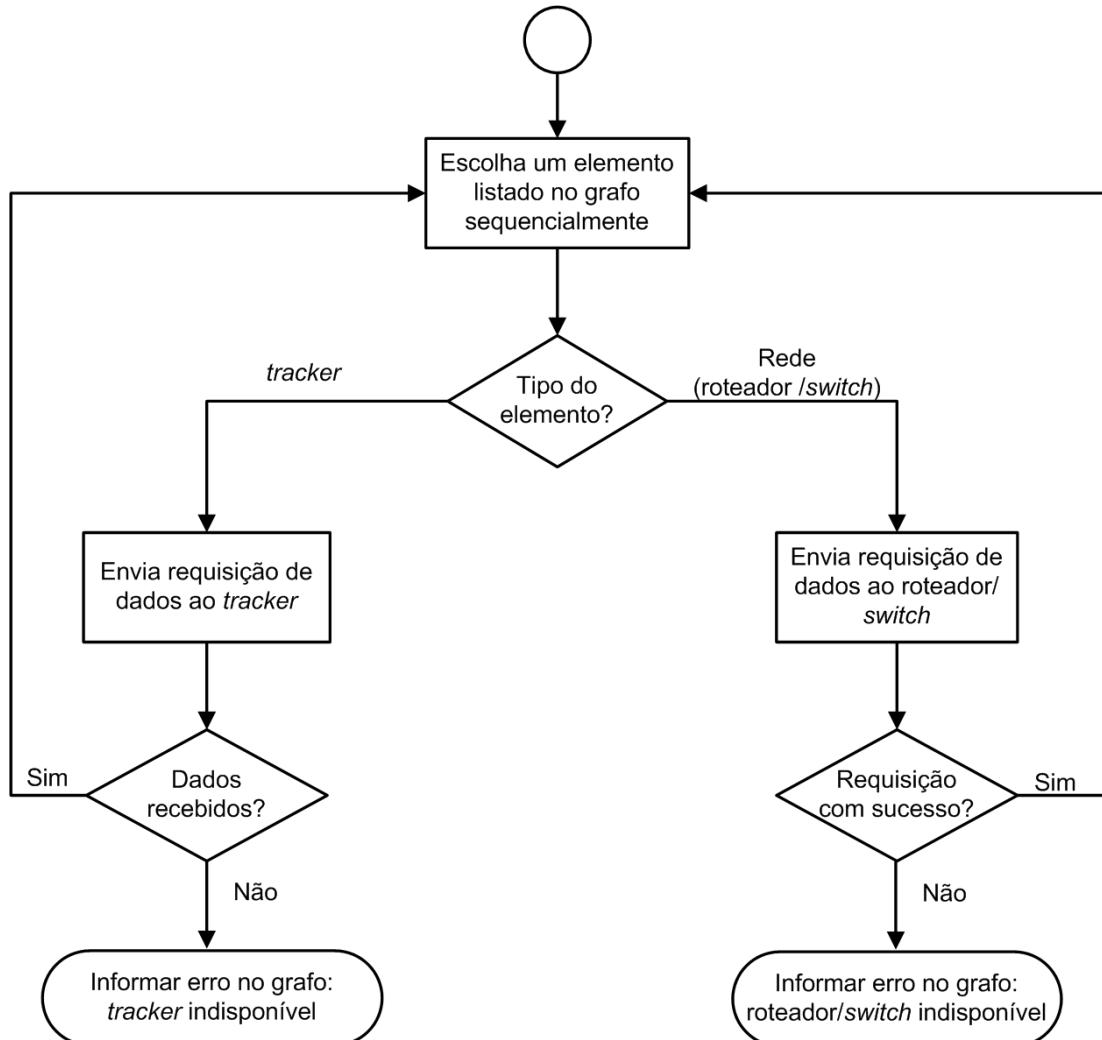


Figura 15: Verificação dos elementos listados no perfil do grafo no *Root Tracker*

A Figura 15 descreve o fluxo de verificação aplicado a cada elemento listado no perfil do *Root Tracker*. Após essa verificação o *Root Tracker* está pronto para realizar suas operações, podendo variar entre os seguintes estados (Figura 14):

- **Estado 1 - Verificação elementos listados no perfil do grafo.** O *tracker* primeiro verifica a existência de todos os elementos listados no grafo, ao mesmo tempo que solicita os dados para poder calcular os pesos das arestas pela primeira vez. Esta verificação é necessária para identificar a existência de erros no perfil do grafo ou a indisponibilidade de algum elemento listado.

- **Estado 2 - Atualização do grafo.** Pode ocorrer de o *Root Tracker* já ter o seu grafo totalmente atualizado mas uma determinada condição (definida no MGP) exigir a aplicação de um outro perfil de grafo no *Root Tracker*. Nessa situação o *Root Tracker* não armazena os dados do grafo anterior e faz uma nova coleta (dados diferentes podem ser solicitados) durante a verificação dos elementos listados no perfil que será aplicado.
- **Estado 3 - Atualização do vértice/nó do grafo.** As atualizações dos dados sobre os elementos de rede é feita periodicamente segundo um intervalo de tempo definido no perfil do grafo. Assim, cabe ao *Root Tracker*, de modo análogo aos *trackers*, fazer as requisições aos elementos de rede. Toda vez que um *tracker* identifica alguma modificação em algum conteúdo ou bloco, ele informa ao *Root Tracker* para que este atualize as suas TC.
- **Estado 4 - Aguardando requisições.** Estado padrão que o *Root Tracker* permanece quando não há nenhuma requisição ou logo após ser inicializado.

4.4 Caso de uso

Conforme descrito no Capítulo 2 os sistemas P2P são desenvolvidos para atender as necessidades de aplicações específicas. Neste contexto, cada aplicação requer métricas específicas para avaliar a eficiência do sistema, sendo que estas métricas usualmente variam de aplicação para aplicação de acordo com o contexto. Por esta razão, a solução proposta possui a característica de poder ser ajustada de acordo com as necessidades específicas, permitindo ao operador da rede modelar o comportamento da rede.

Para demonstrar o comportamento da solução, foi definido um caso de uso no qual estão cobertas as ações para as seguintes situações:

- Obtenção de um conteúdo disponível no mesmo domínio do *peer* requisitante;

- Obtenção de um conteúdo indisponível no mesmo domínio do *peer* requisitante;
- Obtenção de conteúdo com mudanças nas condições da rede durante a sua obtenção; e
- Modificação do comportamento da rede configurada pelo operador da rede durante a obtenção de um conteúdo.

4.4.1 Obtenção de conteúdo

Ator: Usuário final (*peers*):

1. Usuário final executa uma aplicação que emprega a solução proposta para obter o conteúdo;
2. *Peer P* obtém os metadados (arquivos .torrent) pertinentes à solicitação do repositório do sistema;
3. *Peer P* solicita inclusão no *swarm* do conteúdo solicitado e uma lista de *peers* ao *Tracker T* do seu domínio;
4. *Tracker T* verifica *peers* no seu domínio, que possuem o conteúdo solicitado, e elabora uma lista de acordo com os critérios definidos no perfil do grafo;
5. *Tracker T* retorna lista de *peers* a *P*;
6. *Peer P* estabelece uma conexão (usando o protocolo BitTorrent modificado) com todos os *peers* da lista fornecida e obtém o conteúdo;
7. *Peer P* retorna ao passo 2 até obter todo o conteúdo solicitado pela aplicação do usuário.

Caso de exceção: Conteúdo não disponível no sistema:

4. *Tracker T* verifica *peers* no seu domínio que possuem o conteúdo solicitado e elabora uma lista de acordo com os critérios definidos no perfil do grafo;
 - (a) *Tracker T* solicita lista de *trackers* que possuam o conteúdo solicitado pelo *Peer P* ao *Root Tracker*;
 - (b) *Root Tracker* verifica disponibilidade do conteúdo solicitado no sistema;
 - (c) *Root Tracker* retorna mensagem de conteúdo não disponível ao *Tracker T*;
 - (d) *Tracker T* retorna mensagem de conteúdo não disponível ao *Peer P*; e
 - (e) *Peer P* retorna à aplicação a mensagem de conteúdo não disponível.

Caso alternativo 1: Conteúdo não disponível no domínio:

4. *Tracker T* verifica *peers* no seu domínio que possuem o conteúdo solicitado e elabora uma lista de acordo com os critérios definidos no perfil do grafo;
 - (a) *Tracker T* solicita ao *Root Tracker* lista de *trackers* que possuam o conteúdo solicitado pelo *Peer P*;
 - (b) *Root Tracker* verifica disponibilidade do conteúdo solicitado no sistema e elabora lista de *trackers* de acordo com os critérios definidos no perfil do grafo;
 - (c) *Root Tracker* retorna lista de *trackers* para *Tracker T*; e
 - (d) *Tracker T* contacta *trackers* da lista para *Tracker T* e obtém lista de *peers*.

4.5 Limitações da solução proposta

O BitTorrent é classificado quanto ao grau de descentralização como P2P descentralizado híbrido (classificação apresentada na seção 3) por empregar um *tracker* para

organizar a busca do conteúdo e gerenciamento dos *swarms*. Apesar de ser possível ter múltiplos *trackers* para obter um conteúdo, essa especificação ainda encontra-se em fase experimental e deste modo ignorada neste estudo. Sendo assim, o *tracker* é um *Single Point of Failure* (SPF), visto a dependência dos *peers* em relação ao mesmo.

As informações sobre a quantidade de conteúdos disponíveis e *peers* conectados de alguns *trackers* são apresentadas na Tabela 8.

Tabela 8: Dados de *trackers* BitTorrent públicos. Adaptado de: (ISOHUNT, 2012)

url <i>Tracker</i>	servidores	<i>peers</i> ativos	torrents na base
torrentreactor.net	18.604	2.404.716	273.756
www.1337X.org	13.255	2.694.010	85.169
www.archive.org	1 servidor	24.039	774.179
torrents.jamendo.com	1 servidor	39.131	66.388
torrent.ubuntu.com:6969	5 servidores	17.977	1.388
torrents.freebsd.org:8080	1 servidor	404	2.889

Uma abordagem comum empregada por provedores BitTorrent é concentrar todos os metadados (arquivos .torrent) em um repositório central com os conteúdos dispersos em vários *trackers*. É importante ressaltar que cada conteúdo só é gerenciado por um *tracker*, e que nesta situação os conteúdos são dispersos segundo algum critério, não necessariamente pela capacidade de processamento do *tracker*, mas principalmente pela capacidade do enlace que conecta o *tracker* à Internet. Além disso, alguns provedores (e.g., 1337X, torrentreactor) permitem que sejam carregados metadados controlados por *trackers* domésticos e que contribuem para o aumento significativo do número de servidores apresentado na tabela. Existem provedores (e.g., ubuntu, freebsd) que preferem manter poucos ou apenas um servidor porém com mais resiliência, isto é: servidores que possuem um grau de confiabilidade maior e são menos prováveis de ficarem indisponíveis. Desta forma, observando a Tabela 8, pode ser observado que alguns provedores de conteúdo demandam apenas um *tracker* enquanto outros podem demandar milhares.

Analogamente ao BitTorrent, os *trackers* do sistema P2PM proposto também necessitam apresentar características de confiabilidade (resiliência) em função do seu papel essencial para o funcionamento da solução. Se comparadas às operações de um *tracker* BitTorrent com os *trackers* propostos, pode ser afirmado que a exigência de confiabilidade é bem maior na solução proposta. Enquanto que um *tracker* BitTorrent convencional após fornecer a lista de *peers*, e esta conter um *seeder* ou todo conteúdo dispersos nos *leechers*, o *tracker* pode ficar indisponível porque os *peers* não precisam deste para obter todo o conteúdo. Essa mesma situação apresenta problemas na solução proposta. No P2PM, o *tracker* precisa estabelecer contato com os *peers* regularmente para poder gerar as lista de *peers* otimizadas e até mesmo mudar a lista de *peers* para o restante de um conteúdo se as condições (rede ou regras de negócios) não forem mais adequadas. Sendo assim, os *trackers* da solução proposta (P2PM) demandam de soluções adicionais para possuírem um nível adequado de confiabilidade (e.g., redundância, *cluster*).

Outra possível limitação está no fato do tráfego de controle (obtenção das condições dos recursos e aplicação de perfis pelo MGP) compartilhar os mesmos enlaces usados para a troca de dados entre os *peers* e demais serviços em execução na rede. Essa característica pode levar a degradação do desempenho do P2PM, se houver congestionamento nos enlaces, o que pode prejudicar as próprias medições para detectar o problema. Porém, embora não tenha sido desenvolvido na presente proposta, esse problema pode ser resolvido através da configuração de prioridades de tráfego dos enlaces nos equipamentos de rede.

4.6 Trabalhos relacionados

A identificação de trabalhos relacionados foi realizada através de uma busca exaustiva usando mecanismos de busca (Google e Portal da Capes) pelos seguintes termos:

- P2P *locality*;
- *Hierarchical trackers*;
- BitTorrent *optimizations*; e
- BitTorrent *enhancements*.

Os mesmos termos em inglês foram pesquisados em português. O resultado dessa busca foi refinado para apenas os trabalhos que, de alguma forma, possuem similaridades com a solução proposta.

(DOYEN; NATAF; FESTOR, 2005) apresentam uma proposta de hierarquia em vários níveis para agregar *peers* de uma mesma região usando um sistema integrado por meio de várias DHTs que são mantidas por *peers* com diferentes níveis de importância no sistema. A cada *peer* é atribuído um peso que está associado a sua relevância dentro do sistema. São definidas duas camadas: gerenciamento e rede de sobreposição das DHTs. A camada de gerenciamento define como a hierarquia de *peers* é estabelecida de acordo com os pesos de cada *peer*, sendo que este valor é dinâmico, implicando em mudanças na hierarquia. A camada de rede de sobreposição das DHTs é responsável pela inserção, remoção e manutenção dos nós dentro na DHT. A proposta é classificada quanto as suas características de localidade como: critério de formação espacial e a busca/gerência é baseado em servidor estruturada (seção 3.3). Analisando a solução percebe-se que não há qualquer otimização no uso dos recursos da rede, tampouco mecanismos que busquem moldar o tráfego do serviço. Existe uma característica de localidade espacial que é intrínseca das DHTs, porém que fica limitada aos registros existentes em cada segmento. Também não é descrita uma otimização da lista de *peers* em relação ao conteúdo. A ponderação (pesos dos *peers*) está relacionada à importância do *peer* para o sistema como um todo mais do que para atender requisições de um *peer*, i.e., os benefícios obtidos são uma consequência e não uma causa.

Outro trabalho relacionado, que possui similaridades com (DOYEN; NATAF; FESTOR, 2005), é apresentado por (GHANSHANI; BANSAL, 2005). As diferenças entre estes trabalhos é essencialmente o tipo de DHT usado e os critérios de ponderação da importância dos nós. Sendo assim, a análise quanto às características de localidade e otimização são as mesmas de (DOYEN; NATAF; FESTOR, 2005).

Uma dissertação de mestrado correlata é a de (GALLO, 2009), cujo conteúdo será abordado na seção 5.1. Nesta dissertação é descrita uma arquitetura de IPTV que fornece serviços de: transmissão ao vivo usando um sistema *multicast* e um serviço de apresentação deslocada no tempo (*time-shift*) usando uma versão modificada do BitTorrent. Também é apresentado um serviço de localidade espacial, baseado no endereçamento IP, objetivando otimizar as operações P2P. A proposta é classificada quanto a suas características de localidade como: critério de formação espacial e a busca/gerência é baseada em servidor estruturado. Contudo, o sistema P2P empregado é uma modificação do BitTorrent que emprega uma DHT ao invés de *tracker* para gerenciar os *swarms* e fornecer listas de *peers*. Além disso, não há qualquer otimização dos caminhos entre *peer* requisitante e *peers* com conteúdo. Também não há a verificação se os *peers* retornados possuem as partes do conteúdo que são solicitadas.

Existe um trabalho de padronização de um serviço de localidade P2P sendo desenvolvido na IETF, proposta por (PETERSON; GURBANI; MAROCCO, 2010) que foi influenciado por (XIE et al., 2008). Este trabalho busca resolver as questões de localidade através da utilização de um *tracker* mais sofisticado que possui algum conhecimento sobre a topologia da rede e *peers*. A proposta é classificada quanto a suas características de localidade como: baseado em interesse por conteúdo e busca/gerência é baseada em servidor não estruturado. O *tracker* é identificado por uma solução similar a empregada para descobrir servidores *Simple Mail Transport Protocol* (SMTP) usando um serviço de *Domain Name System* (DNS), sendo que (XIE et al., 2008) propõe a criação de um tipo de registro especial no sistema de DNS para identificar *trackers*. O cerne da

ideia é encaminhar os *peers* para o *tracker* que está gerenciando o conteúdo solicitado, agrupando todos os *peers* do sistema que estão interessados no conteúdo. Dessa forma, busca-se refinar a lista de *peers* fornecida a um *peer* requisitante pela identificação de quais *peers* estão em uma mesma região da rede usando os seus respectivos endereços IP. Embora a proposta preveja vários *trackers*, os mesmos não estão organizados segundo uma hierarquia, operando de forma autônoma na otimização das listas de *peers*. No entanto, não há otimização dos caminhos entre *peer* requisitante e *peers* com conteúdo e nem a verificação se os *peers* retornados possuem as partes do conteúdo que são solicitadas. Não obstante, a proposta descrita em (XIE et al., 2008) oferece uma otimização quanto ao uso dos recursos da rede ao preferir incluir na lista os *peers* que estão na mesma região do *peer* solicitante. Apesar dos seus méritos, tal proposta não atende adequadamente os requisitos das redes gerenciadas definidas nessa tese. Assim, em cenários que envolvam a entrega de conteúdo via P2P como VoD, onde o operador deve assegurar uma taxa de *download* (e, consequentemente, a continuidade da reprodução do conteúdo), a simples implantação das abordagens descritas nesses trabalhos pode não ser o suficiente e também não explorar todos os recursos disponíveis para otimização da rede.

4.7 Considerações do Capítulo

A solução apresentada neste capítulo propõe a otimização das listas de *peers* com base nas condições da rede e regras de negócio para otimizar os recursos da rede. Embora experiências com protocolos tipo OSPF já mostraram que a configuração de rotas baseadas em informações das condições da rede pode gerar resultados indesejados, na solução apresentada não são alteradas quaisquer tabelas de roteamento em função do tráfego. A otimização é realizada exclusivamente através da lista de *peers* retornada pelo *tracker*, que pode incluir o caminho para encaminhamento através de *Source Routing* caso haja mais de uma rota possível. Contudo, sabe-se da dificuldade em avaliar

soluções que reúnem aspectos de redes *overlay* e redes físicas.

A proposta apresentada de “arquitetura usando *trackers* hierárquicos para localidade em redes *peer-to-peer* gerenciadas” possui uma organização relativamente simples que proporciona um serviço de localidade espacial baseado em uma arquitetura de servidores estruturados. Esta simplicidade é uma característica importante para a escalabilidade do sistema, visto que sistemas complexos usualmente são mais difíceis de escalar.

A eficiência do serviço de localidade proposto é diretamente dependente da corretude dos dados usados nos grafos. Então, é necessário manter estes dados atualizados a fim de evitar que um *tracker* retorne uma lista de *peers* inadequada.

O uso do protocolo BitTorrent como base à solução agrega um sistema P2P que já foi amplamente testado quanto a sua escalabilidade e confiabilidade, evitando a especificação de um novo protocolo onde estes aspectos precisariam ser atestados novamente. Adicionalmente, o uso do BitTorrent também proporciona uma base de comparação consistente para aferir os níveis de otimização resultantes do uso da solução proposta nesta tese.

5 AMBIENTE DE EXPERIMENTAÇÃO

A solução proposta no Capítulo 4 pode ser empregada para diferentes tipos de aplicações de vídeo que consomem dados obtidos por meio de redes P2P. Visando avaliar a eficiência da solução proposta, foi desenvolvido um protótipo baseado no sistema de IPTV com arquitetura de distribuição híbrida (*Multicast* e P2P) descrito em (GALLO et al., 2009).

Um dos diferenciais deste sistema é o serviço de suporte à apresentação de conteúdo deslocado no tempo (aqui chamado de *time-shift*) que emprega P2P para obter o conteúdo de vídeo. O serviço de *time-shift* consiste essencialmente em apresentar um conteúdo que já foi apresentado anteriormente, como por exemplo repetir um lance em uma partida de futebol. Deve ser observado que, quando isto acontece, a linha do tempo é deslocada para o passado e o conteúdo posterior ao momento retrocedido (i.e., seguindo a continuidade do vídeo) é obtido via P2P.

Este sistema será descrito brevemente, limitando-se aos aspectos necessários para entender as modificações introduzidas para testar a solução descrita na presente tese.

5.1 Sistema de IPTV baseado em DHT

A principal ideia deste sistema de IPTV é usar os conteúdos transmitidos via *streaming (multicast)* para armazenamento em *cache*, evitando retransmissões desnecessárias dos conteúdos quando eles são solicitados a partir do serviço de VoD ou *time-shift*. Um pressuposto a respeito do *streaming* é o emprego de IP *Multicast*, uma premissa

razoável em redes gerenciadas que podem suportar serviços que não são suportados em redes públicas como a Internet. (e.g., redes de provedores que estão interessados em explorar sua infraestrutura para oferecer serviços de TV sobre IP). Este Sistema de IPTV permite que os recursos disponíveis nos equipamentos dos usuários finais (e.g., computadores ou *set-top-boxes*) possam ser usados para auxiliar na distribuição do conteúdo. Por meio desta abordagem, os usuários poderiam, por exemplo, permitir o uso de seus recursos ociosos em troca de um serviço mais barato ou uma garantia de melhor QoE (GALLO, 2009).

No protótipo do sistema de IPTV baseado em DHT, o uso do serviço de P2P somente ocorre nas operações de *time-shift*, visto que o serviço de VoD foi apenas especificado; no entanto, como o serviço de VoD usa os mesmos mecanismos empregados no *time-shift*, esta não é uma limitação do sistema mas sim uma decisão de implementação. A Figura 16 apresenta uma visão geral deste sistema de IPTV.

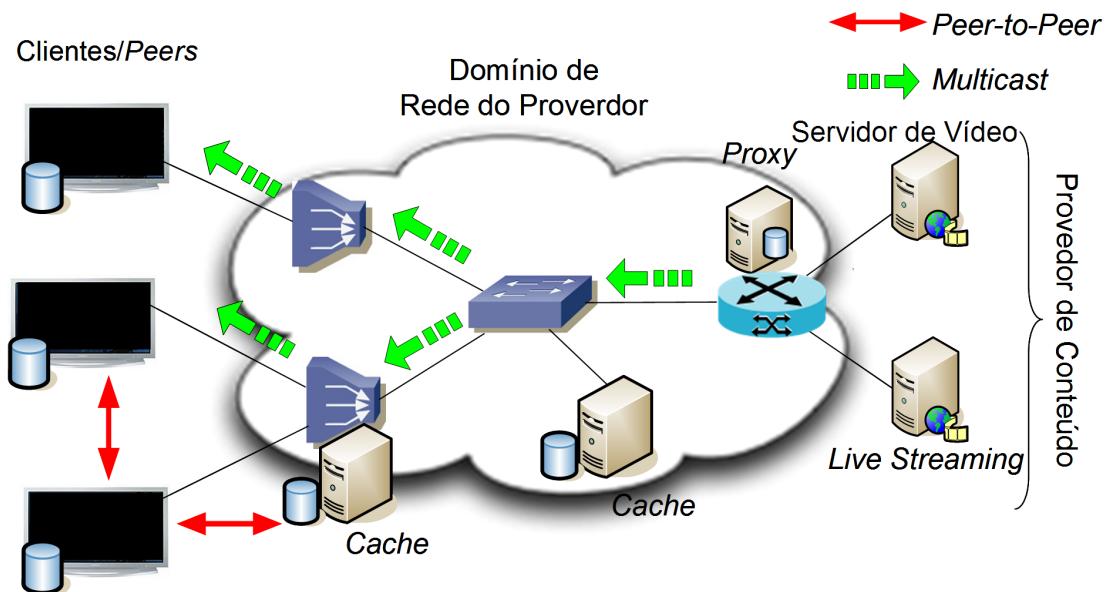


Figura 16: Visão geral do sistema de IPTV baseado em DHT. Adaptado de: (GALLO et al., 2009)

Uma descrição dos elementos apresentados na Figura 16 (GALLO, 2009):

- **Proxy:** módulo responsável por receber o conteúdo que será disponibilizado no

sistema e retransmiti-lo na rede através de IP *Multicast*. Este conteúdo pode ser disponibilizado a partir de um rede externa, como por exemplo um *streaming* de vídeo *unicast* gerado por um provedor de conteúdo. O *Proxy* também realiza a conversão deste conteúdo do formato de *streaming* para o formato P2P do BitTorrent, permitindo seu posterior uso pelo serviço de *time-shift*. Sendo assim, ele gera tanto os metadados (arquivos .torrent) como arquivos locais com o vídeo (e.g., arquivos do tipo AVI).

- ***Cache***: módulo implantando em posições estratégicas da rede para armazenar cópias locais do conteúdo transmitido através de IP *Multicast*. Este conteúdo é convertido para P2P do tipo BitTorrent e sincronizado com o *Proxy* para assegurar que os arquivos gerados no *Cache* estejam organizados da mesma forma que no *Proxy*.
- ***Clientes/Peers***: módulo que tem a capacidade de entrar em um grupo *multicast* para reproduzir conteúdo do tipo transmissão ao vivo, ou obter partes do conteúdo já transmitidas através do serviço de *time-shift*. No serviço de *time-shift* o *peer* solicita as partes do conteúdo (chamadas de blocos) na ordem em que elas foram transmitidas.
- ***Servidor de Vídeo***: aplicação *VideoLan Client* (VLC) padrão que tem como função fazer a ingestão de conteúdo *multicast* no sistema.

A parte P2P deste sistema é usada para obter o conteúdo consumido pelo serviço de *time-shift*. O conteúdo do P2P é organizado em uma DHT do tipo CHORD (STOICA et al., 2001), estando distribuída em todos os elementos do sistema que possuem a capacidade de comportar-se como um *peer* (GALLO et al., 2009). Isto inclui todos os Clientes assim como os módulos de *Cache* e o *Proxy*, sendo que estes dois últimos elementos possuem um comportamento de *seeder*, i.e., fornecem conteúdo para a rede P2P e não obtém conteúdos usando P2P. Uma vez identificados os *peers* com o conteúdo, o Cli-

ente aplica um mecanismo de localidade espacial baseado no endereçamento IP para selecionar prioritariamente os *peers* mais próximos e obter o conteúdo continuamente até que o usuário o interrompa.

Os sistemas BitTorrent não foram originalmente projetados para fornecer conteúdo do tipo transmissão ao vivo, sendo planejados para distribuição de algum conteúdo pré-armazenado. No caso da transmissão ao vivo, o *streaming* de vídeo não pode originalmente ser convertido e disponibilizado ao mesmo tempo. A solução adotada para esse problema é gerar pequenos arquivos à medida que o conteúdo do *streaming* é recebido. Esses pequenos arquivos então são preparados para distribuição via BitTorrent, o que envolve a geração de ID do Conteúdo, criação de metadados, etc. A Figura 17 exemplifica esta solução aplicada a um segmento de 28 segundos de *streaming* de um conteúdo *X*, que é dividido em blocos de 2 segundos.

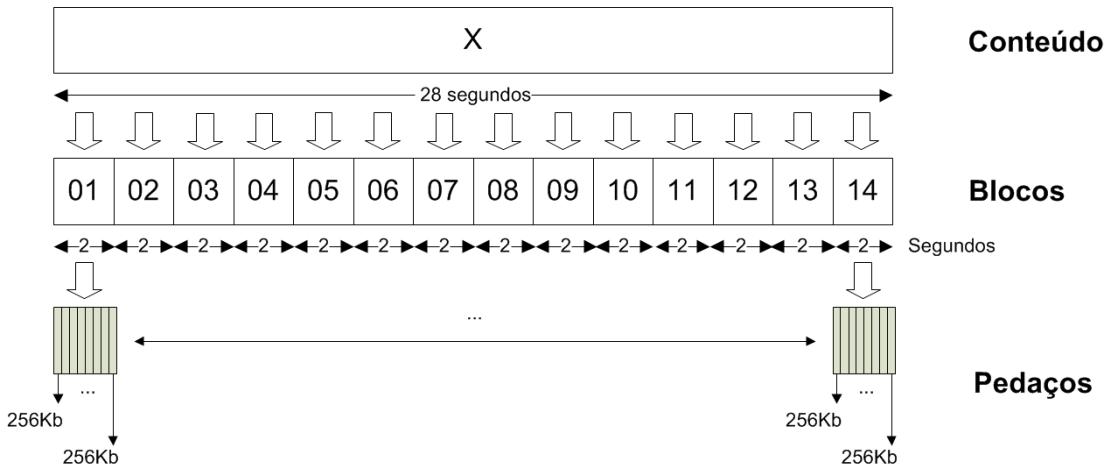


Figura 17: Transformação do conteúdo em para uso no BitTorrent

Cada bloco gerado é preparado para distribuição BitTorrent de forma independente, de modo que no exemplo apresentado são gerados 14 arquivos .torrent. O tamanho do bloco é personalizável e define a granularidade da operação de *time-shift*, visto que o usuário pode apenas retroceder para um bloco específico dentro da linha do tempo. Cada bloco é composto de pedaços, da mesma forma que no BitTorrent.

No entanto, uma modificação feita no protocolo BitTorrent para esse sistema de IPTV refere-se à ordem de obtenção dos pedaços. Enquanto no BitTorrent padrão esta ordem é definida pelos algoritmos *Raros Primeiro* e *Primeiro Pedaço Aleatório* (descritos na seção 2.3), tal abordagem não é adequada para arquivos de vídeo onde o conteúdo necessita ser reproduzido sequencialmente. Assim, no sistema IPTV o algoritmo de busca foi modificado para obter os pedaços sequencialmente. Além disso, o Cliente fica buscando sequencialmente os metadados dos blocos seguintes tendo como objetivo a reprodução do vídeo sem interrupções.

5.2 Sistema IPTV com serviço de localidade

A principal diferença entre o sistema apresentado por (GALLO et al., 2009) está no serviço de localidade: a solução P2PM (Capítulo 4) usa uma arquitetura de *trackers* hierárquicos ausente no trabalho de (GALLO et al., 2009). Especificamente, a nova versão do sistema IPTV emprega a arquitetura de *trackers* descrita na seção 4.3 ao invés de DHT. Um dos principais benefícios desta nova abordagem é que, no caso de uso de uma DHT, o cliente precisa inicialmente descobrir os *peers* que possuem os conteúdos solicitados através de uma requisição, o que pode implicar na comunicação com vários *peers*; isto ocorre porque o sistema original apresenta apenas uma DHT que concentra a informação de todos os *peers* de forma distribuída, introduzindo problemas de escalabilidade devido ao fato da busca ser sempre completa.

Além disso, o sistema foi portado para permitir tanto o serviço de *time-shift* como o de VoD. No modo VoD os conteúdos são inseridos no sistema através do Módulo *Proxy*, que prepara o conteúdo para distribuição P2P.

Os módulos do sistema de IPTV com serviço de localidade proposto são, portanto:

- ***Proxy***: tem o mesmo papel do *Proxy* no sistema original, mas também é capaz de tratar conteúdo de VoD.

- ***Cache***: equivalente ao módulo de *Cache* no sistema original, tendo como funcionalidade adicional a capacidade de armazenar conteúdo para o serviço de VoD, que precisa ser copiado do *Proxy*.
- ***Tracker***: módulo responsável por gerenciar as operações P2P dentro do seu domínio e prover o serviço de localidade. Além disso, o *tracker* também coleta as informações dos elementos de rede para atualizar o seu grafo e monitora a necessidade de substituir o grafo atual por outro perfil.
- ***Root Tracker***: módulo responsável por gerenciar os *trackers* abaixo na sua hierarquia e auxiliar no serviço de localidade entre domínios. Além disso, o *tracker* também coleta as informações dos elementos de rede do seu domínio para atualizar o seu grafo e monitora a necessidade de substituir o grafo atual por outro perfil.
- **Módulo Gerenciador de Perfil (MGP)**: módulo responsável por orquestrar os perfis em todo o sistema e configurar as regras que irão reger e ativar o processo de substituição de perfis.
- **Clientes/Peers**: módulo que tem a capacidade de entrar em um grupo *multicast* para reproduzir conteúdo do tipo transmissão ao vivo, ou obter partes do conteúdo já transmitidas através do serviço de *time-shift*. No serviço de *time-shift*, o Cliente solicita as partes do conteúdo (também chamadas de blocos) na ordem que elas foram transmitidas. No serviço VoD, o Cliente faz uma operação análoga ao *time-shift*, porém seleciona o ID do Conteúdo com base em uma escolha no catálogo de um *Electronic Programming Guide* (EPG).

Nos serviços de VoD e *time-shift* o *Cache* é equivalente a um Cliente com conteúdo já disponível (*seeder*) e pode ser representado como um *Cliente/Peer*. Por outro lado, O *Proxy* acumula função similar de *Cliente/Peer (seeder)*, mas continua possuindo as suas atribuições de conversão de conteúdo e *multicast*.

5.2.1 Implementação

Todo o sistema foi desenvolvido usando a linguagem de programação Java. Foram empregadas várias *Application Programming Interface* (API)s para manipular o conteúdo de vídeo e usar o BitTorrent, sendo as principais:

- **Java Media Framework (JMF)**¹: Usada para receber os pacotes *multicast* e extrair informações do cabeçalho para realizar a sincronia entre o *Proxy* e os *Caches*, quando é realizada a conversão de *streaming multicast* para P2P;
- **Java BitTorrentAPI**²: usada como base para desenvolver os *trackers* e *Root Tracker* e também a versão modificada do BitTorrent usada no sistema IPTV;
- **Java bindings for VideoLan Client (jVLC)**³: usada para realizar as transmissões *multicast* no *Proxy* e nos Clientes tanto para receber o conteúdo *multicast* como reproduzir o conteúdo obtido através do P2P; e
- **JGraphT**⁴: usada para criar os grafos, manipular os nós/vértices e calcular o menor caminho entre eles.

A comunicação de controle entre *Proxy*, *Trackers*, *Root Tracker* e MGP é realizada através de *web services*. Os *web services* foram escolhidos para padronizar a comunicação entre os elementos do sistema usando um formato homogêneo de mensagens. Contudo, os *trackers* precisam obter informações de elementos de rede, como por exemplo, dos roteadores, que usualmente não possuem acesso através de *web services*. Para isso, a opção mais padronizada, prática e amplamente difundida de obter dados de uso é através de requisições usando o protocolo SNMP. Sendo assim, a obtenção dos dados dos elementos de rede para atualizar os grafos (*Trackers* e *Root Tracker*)

¹<http://www.oracle.com/technetwork/java/javase/download-142937.html>

²<http://sourceforge.net/projects/bitext/>

³<http://javaplayer.sourceforge.net/doc/vlc/JVLC.html>

⁴<http://www.jgrapht.org/>

é realizada através do uso de protocolo SNMPv2 que solicita um dado na árvore MIB dos equipamentos. As informações disponíveis em uma MIB podem sofrer variação entre diferentes modelos de equipamento e até mesmo software/firmware, sendo recomendado empregar dados que estejam em uma MIB definida e regulamentada por uma *Request for Comments* (RFC).

Os sistemas IPTV possuem critérios que são relacionados ao provedor de rede e outros relacionados aos usuários (Clientes). O objetivo do sistema de localidade aqui proposto é atender ambos conjuntos de critérios. Assim, foram definidos dois critérios para avaliar o serviço de localidade:

- **Uso dos enlaces de rede:** montante de tráfego gerado em cada enlace para atender uma solicitação de um Cliente, desde o momento da solicitação até a obtenção de todo o conteúdo.
- **Tempo de inicialização:** tempo decorrido do exato instante em que a solicitação é feita no Cliente até o primeiro quadro do vídeo ser exibido na tela.

Estes dados permitem avaliar como o serviço de localidade impactou no uso dos enlaces (algo que é do interesse do provedor de rede) e o tempo de inicialização do Cliente (algo que contribui para a QoE do usuário final). O serviço de localidade não seria tão útil para um provedor se ele otimizasse o uso dos recursos da rede mas degradasse o tempo de inicialização ao ponto de torná-lo incômodo para o Cliente. Percebe-se então, a prudência necessária para definir as informações e os critérios nos perfis de grafo.

Conforme discutido na seção 4.3, a solução proposta permite que o cálculo de um peso w da aresta de um grafo seja personalizada de acordo com os critérios desejados. Na implementação realizada neste protótipo foi adotado um critério baseado na priorização dos enlaces, fazendo com que o peso calculado seja susceptível a mudanças de acordo com as variações da banda disponível no enlace. Isto pode ser realizado de

várias formas sendo aqui adotada uma fórmula simplificada para facilitar a relação dos critérios:

$$w = \frac{1}{(Prioridade * BandaDisponivel)}$$

O valor de *Prioridade* é definido estaticamente pelo operador da rede e representa a prioridade na seleção de um enlace em um intervalo de valores entre [1 – 100], onde o valor 1 representa a mais alta prioridade. Deste modo, o operador de rede pode configurar o valor de *Prioridade* para evitar (ou reduzir) o tráfego de rede em um conjunto de enlaces, para diminuir o custo das operações de rede, ou, mesmo, para evitar o tráfego em enlaces problemáticos. Nos experimentos realizados, foi escolhido *Prioridade* = 80, como valor padrão para tornar os pesos das arestas mais sensíveis a flutuações do valor de *BandaDisponivel*. O valor de *BandaDisponivel* é coletado pelos *Trackers* a partir de requisições SNMP (medidas em Kbps) enviadas aos roteadores. A taxa de atualização escolhida nos experimentos foi de 15 segundos e, portanto, este também é o intervalo em que o peso nas arestas dos grafos (*Trackers* e *Root Tracker*) são atualizados.

A lista de *peers* elaborada pelos *Trackers* é gerada empregando-se o Algoritmo de Dijkstra localmente nos domínios. Todos os perfis de grafo são armazenados em arquivos *Extensible Markup Language* (XML) segundo os padrões da API JGraphT. Como exemplo, o arquivo relativo ao perfil de grafo do *TrackerI* pode ser encontrado no Apêndice B.

5.3 Configuração do Ambiente de Teste

O principal objetivo de implementar a solução proposta no cenário de IPTV é o de mostrar como o uso dos recursos de uma rede gerenciada pode ser otimizado. O protótipo foi instalado no ambiente de teste (*testbed*) representado na Figura 18, sendo

que diversos experimentos foram executados para medir o uso dos recursos da rede e desempenho da aplicação de vídeo.

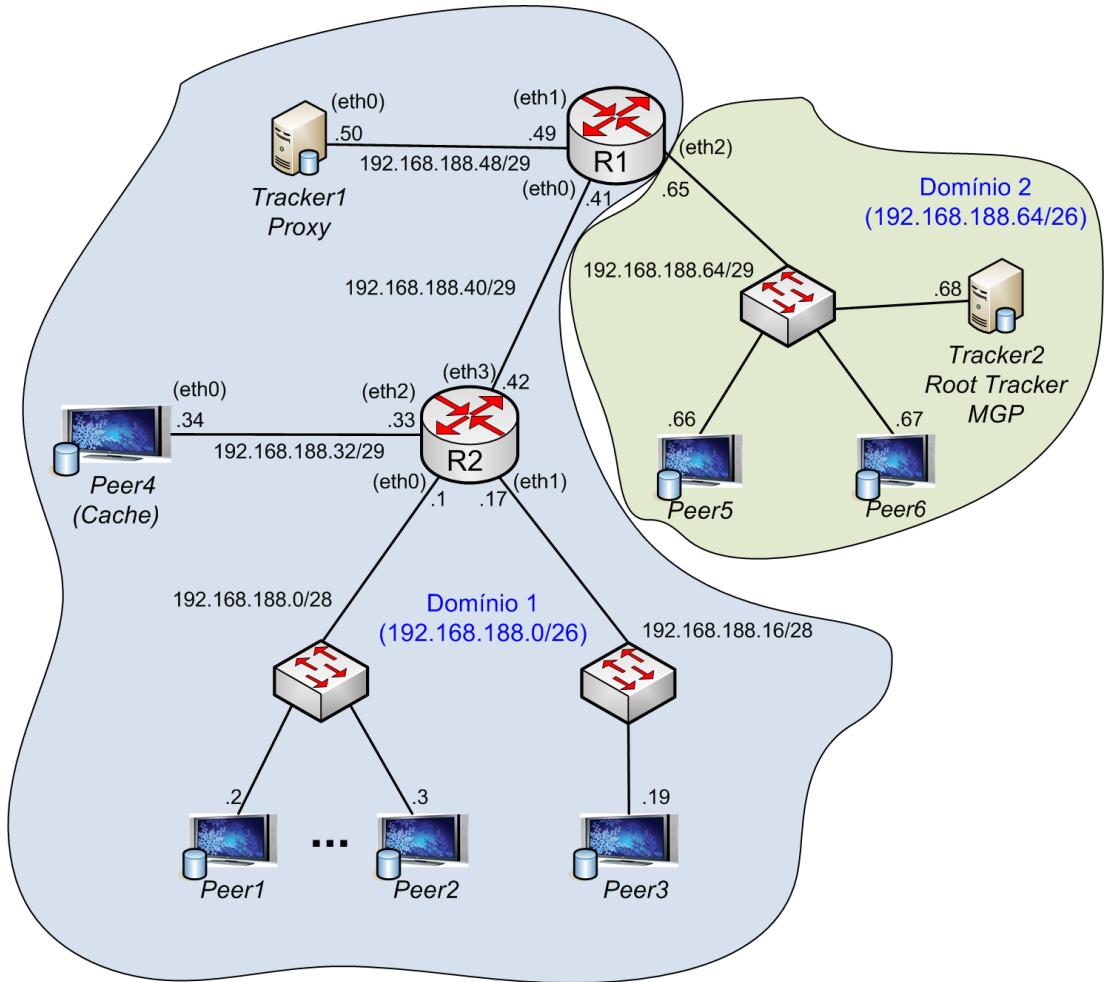


Figura 18: Disposição dos elementos do sistema IPTV e topologia da rede

No cenário adotado para os experimentos (Figura 18), o *Tracker1* e o *Proxy* foram instalados no mesmo computador para simplificar a operação do sistema. De modo similar, o MGP foi disposto junto com o *Root Tracker* no mesmo equipamento. Além disso, a topologia do ambiente busca reproduzir diferentes níveis de rede (Rede de Borda, Roteadores Regionais e Rede WAN) de modo similar ao apresentado na Figura 9. Todos os recursos empregados neste ambiente de teste foram alocados exclusivamente para tais experimentos, evitando a inserção de tráfego externo que pudesse interferir nas medições. Diante da indisponibilidade de roteadores reais, foi adotada a

solução de roteador por software (*software router*) empregando-se o software Vyatta⁵, que foi selecionado por sua popularidade e confiabilidade perante a comunidade científica e em organizações (DOBRESCU et al., 2010). Descrição do hardware/software/rede configurado nos equipamentos é:

- **Software:**

- Sistema operacional do *Proxy*, *Cache*, *Trackers*, *Root Tracker*, MGP e todos os Clientes: foi realizada uma instalação limpa do Ubuntu 8.10. Apesar de existirem versões mais recentes deste S.O., particularidades do jVLC (principalmente a compatibilidade de bibliotecas) forçaram o uso desta versão em específico.
- Sistema operacional dos roteadores: Vyatta Community Edition versão 5.

- **Rede:** Foi empregado um *switch* 3Com modelo 2924-SFP Plus (24 ports 1Gb) dedicado para este ambiente de teste e todas as máquinas foram interconectadas através de *Virtual Local Area Network* (VLAN) e isoladas de acordo com a rede a que pertencem (isolamento do tráfego). Sendo usados os roteadores para conectar essas diversas VLAN através de roteamento estático.

- **Hardware:** todos os computadores utilizados possuem a mesma configuração: Pentium QuadCore 2.4 GHz, 4 GiB RAM, 500 GiB Hard Disk e 4 interfaces Ethernet (1Gb).

Os relógios de todos os computadores foram também ajustados para ficarem sincronizados através do uso do *Network Time Protocol* (NTP), permitindo armazenar as medições localmente em cada computador durante os experimentos. Após cada execução de um experimento ser finalizada, os dados eram copiados via *shellscrip*t para o computador onde está o *Proxy/Tracker1*, e eram, então, analisados. O processo de captura de dados/*logs* sobre o uso do sistema é composto de duas partes:

⁵<http://www.vyatta.com>

1. **Logs do Sistema de IPTV:** em vários pontos críticos do sistema (e.g., quando Cliente faz uma requisição de conteúdo), o módulo registra a solicitação em arquivo texto que é armazenado localmente. Estas informações são usadas essencialmente para calcular o tempo de inicialização.
2. **Arquivos de Dump:** antes de cada experimento começar a ser realizado, é executada uma fase de controle na qual o aplicativo TCPDump⁶ é inicializado em cada computador do sistema para coletar todos os dados das interfaces de rede em uso. Somente após esta fase é que são iniciadas as requisições de conteúdo.

A análise dos dados/*logs* foi feita através de um *script* que calcula resultados e cria um arquivo tipo *Comma-Separated Values* (CSV) com o resumo dos mesmos. No caso dos arquivos de *dump*, estes são filtrados usando o aplicativo tshark⁷, versão de linha de comando do Wireshark; em seguida, um *script* os analisa de forma análoga aos logs e também publica o resumo em um arquivo CSV.

Analizar o comportamento do sistema apenas após os experimentos é algo que tornaria moroso a sua depuração e ajuste nos perfis dos grafos. Por esta razão, foi desenvolvida uma interface de monitoração que permite verificar o estado do sistema em tempo real, cuja tela principal é mostrada na Figura 19.

⁶<http://www.tcpdump.org/>

⁷<http://www.wireshark.org/docs/man-pages/tshark.html>

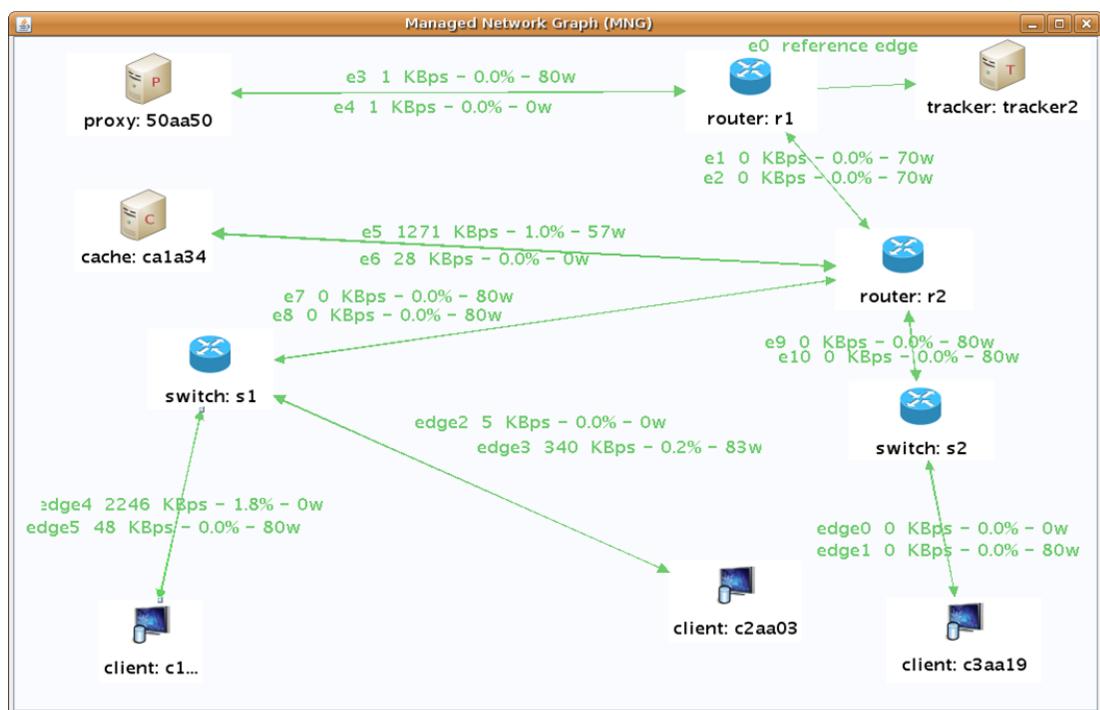


Figura 19: Interface de monitoração do sistema em tempo real

A interface de monitoração possibilita o acompanhamento da execução dos experimentos por meio das várias informações disponíveis para leitura (e.g., lista de *peers* retornada pelo *tracker*, uso dos enlace, etc.). Adicionalmente, as linhas que representam as arestas reagem a mudanças das condições da rede de modo que sua espessura aumenta proporcionalmente ao tráfego e as cores variam entre verde (enlace não está sobrecarregado), amarelo (enlace com uma certa carga) e vermelho (enlace sobrecarregado).

Também foi criada uma interface para permitir a monitoração das arestas do sistema, apresentada na Figura 20.

Graph and Network Information Table (GNIT)						
Edge ID	Source ID	Target ID	Weight	Priority	Available Bandwidth (Kbps)	
e0	r1	tracker2	0	0.0	0	124999
e1	r1	r2	70003	70.0	124993	
e2	r2	r1	70062	70.0	124558	
e3	50aa50	r1	84411	80.0	124813	
e4	r1	50aa50	0	0.0	124894	
e5	ca1a34	r2	46051	45.0	124997	
e6	r2	ca1a34	0	0.0	124153	
e7	r2	s1	88466	80.0	124153	
e8	s1	r2	88466	80.0	124999	
e9	r2	s2	80001	80.0	124999	
e10	s2	r2	80001	80.0	124999	
edge0	s1	c2aa03	0	80.0	124997	
edge1	c2aa03	s1	81106	80.0	124889	
edge2	s2	c3aa19	0	80.0	124999	
edge3	c3aa19	s2	80001	80.0	124999	
edge4	s1	c1aa02	0	80.0	123785	
edge5	c1aa02	s1	82105	80.0	124789	

Figura 20: Interface da monitoração das arestas do sistema

A interface do Sistema P2PM possui várias janelas, sendo que a Figura 20 apresenta a sua tela principal. Diversas outras informações são apresentadas nessa interface, como: lista de *peers* retornados pelo *tracker*; quantidade de blocos contidos no *buffer* do Cliente; histórico do uso de cada enlace individualmente; etc. Outras figuras de exemplos estão disponíveis no Apêndice C.

5.4 Limitações da implementação

No decorrer do desenvolvimento do serviço de localidade e aprimoramento do Sistema IPTV foram identificadas algumas limitações de operação. Estas limitações são em grande parte devido ao fato do protótipo ter sido desenvolvido com a finalidade de ser uma prova de conceito do serviço de localidade e não um sistema de produção.

Algumas limitações têm destaque:

- **Uso do Source Routing:** A API do JavaBitTorrent possui uma implementação muito simples, porém seu código apresenta pequenos erros e reduzido grau de otimização. Vários desses problemas foram identificados por (GALLO, 2009) e corrigidos na versão atual do Sistema IPTV com o serviço de localidade. Con-

tudo, não foi possível implementar o *Source Routing* por limitações nesta API quanto a manipulação do cabeçalho IP. Para contornar este problema, a topologia do ambiente de teste foi configurada no formato de uma árvore, de modo que haja apenas um caminho possível quando um *peer* necessita comunicar-se com outro.

- **Inserção de conteúdo VoD no sistema:** Atualmente o processo de preparação de um conteúdo VoD necessita ser feito manualmente no *Proxy*. Isto não afeta o desempenho do sistema, mas tal procedimento manual torna a mudança dos conteúdos disponibilizados mais trabalhosa.
- **Uso de diferentes *codecs* de vídeo:** Uma limitação no jVLC impede o uso de vários *codecs*, limitando o uso ao *codec* MPEG2.

Vários dos problemas apontados podem ser resolvidos através de melhorias no código e novos desenvolvimentos. Porém, como o foco da presente tese está no serviço de localidade e este pode ser avaliado mesmo com essas limitações presentes, tais melhorias não foram desenvolvidas nesse momento.

5.5 Considerações do Capítulo

O uso da solução de localidade descrita nessa tese para um sistema de IPTV permite avaliar a aplicabilidade da mesma e definir aspectos concretos de desempenho que foram abstraídos na proposta. Questões como critérios para calcular os pesos das arestas e métricas de avaliação de eficiência puderam ser definidos, assim como a abordagem para coleta destes dados nos elementos do sistema.

O fato de já existir o Sistema IPTV baseado em DHT facilitou o desenvolvimento do Sistema IPTV dotado do serviço de localidade proposto, visto que várias partes do código puderam ser reaproveitadas. Contudo, a existência de problemas no código e

limitações das API produziram como consequência uma implementação voltado para o objetivo de ser prova de conceito, precisando ser melhorada para o seu emprego num ambiente de produção.

A configuração do ambiente de testes apresentou vários desafios devido às limitações com relação aos equipamentos disponíveis e à natureza dos experimentos executados. Finalmente, o poder de processamento dos equipamentos e a quantidade de interfaces de rede foram decisivos para o sucesso do ambiente de testes: o primeiro permitiu que vários módulos pudessem ser executados e agrupados em um mesmo computador, criando-se um cenário adequado para os experimentos; já o segundo permitiu que os *software routers* fossem configurados nesses equipamentos sem a necessidade do uso de interfaces de rede virtuais, que poderiam causar uma distorção no desempenho dos enlaces dos roteadores devido a acumular dois tráfegos distintos em uma mesma interface.

6 EXPERIMENTOS & ANÁLISE DOS RESULTADOS

Os experimentos apresentados neste capítulo foram elaborados para analisar como a solução proposta trata os aspectos de localidade, otimização no uso dos recursos da rede e políticas de uso desses recursos. Os requisitos, definidos na seção 4.1, são também analisados a fim de identificar os pontos satisfeitos pela solução proposta.

A Figura 21 apresenta uma simplificação da Figura 18, na qual foram excluídas as informações sobre endereçamento IP e incluídas as identificações dos enlaces para facilitar a análise dos resultados dos testes.

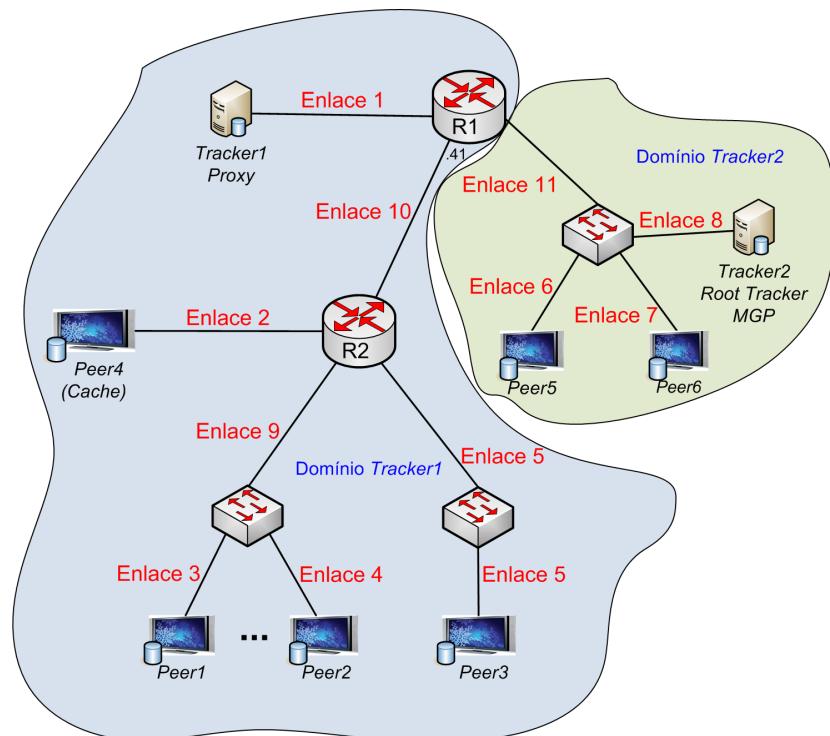


Figura 21: Identificação dos enlaces do *testbed*

Na Figura 21 dois enlaces foram nomeados com o mesmo identificador (Enlace 5). Essa nomeação foi adotada para simplificar a apresentação dos resultados, já que o tráfego entre o roteador *R2* e *switch* será sempre o mesmo tráfego que entre este *switch* e o *Peer3*.

São empregadas as seguintes métricas para avaliar o desempenho do serviço de localidade:

- **Tempo de inicialização:** segundo (GREENGRASS; EVANS; BEGEN, 2009; MANZATO; FONSECA, 2010) o tempo máximo de espera que um usuário tolera para iniciar a visualização de um conteúdo em um sistema IPTV é de aproximadamente 5 segundos, enquanto em sistemas de TV a cabo tradicional este tempo é de dois segundos. O tempo de inicialização é aqui definido como a latência entre a requisição do conteúdo e o início de sua reprodução na tela. Este tempo é aferido em cada execução do experimento através de mensagens de *log* do sistema que registram o tempo exato da requisição e o tempo em que a exibição é iniciada.
- **Uso dos enlaces de rede:** a eficiência no uso dos enlaces é avaliada em termos qualitativos e quantitativos. A avaliação qualitativa é feita através da identificação dos enlaces utilizados em cada teste, enquanto a quantitativa é obtida através do montante de dados trafegados por enlace. O sistema é dito eficiente quando busca o máximo de conteúdo através de enlaces próximos, evitando o acesso a *backbones* Internet. Tanto os aspectos qualitativos como quantitativos desta métrica são aferidos através de arquivos de *dump* coletados nos elementos do sistema. Tais arquivos são então analisados por *scripts* que computam a quantidade de dados trafegada por enlace e identificam os enlaces utilizados.
- **Tempo médio de obtenção dos blocos:** O Sistema IPTV adota uma variação do protocolo BitTorrent, criando vários torrents para um conteúdo. No sistema

apresentado, cada bloco possui dois segundos do conteúdo. Esta métrica foi adotada para identificar o tempo de obtenção do conteúdo. Para os blocos de dois segundos é necessário que o tempo para sua obtenção seja abaixo de dois segundos (tamanho do bloco) evitando-se, assim, a interrupção na reprodução do conteúdo no cliente. O tamanho de dois segundos também foi escolhido para que o armazenamento (*buffer*, principalmente do primeiro bloco do conteúdo) não aumente a latência do tempo de inicialização, visto que é necessário obter um bloco completo para iniciar a reprodução. A informação sobre o tempo de obtenção de blocos é aferida pelo sistema que regista os *timestamps* das operações nos *logs* do Sistema IPTV, registrando o tempo de obtenção de cada bloco. No final das execuções dos testes, é realizada a compilação destes dados para obter a média geral.

- **Interrupção da reprodução do conteúdo:** o tempo médio de obtenção dos blocos fornece uma visão geral do desempenho. Porém, dada a natureza dinâmica do serviço de localidade, podem haver desvios nesta média devido às mudanças na condição da rede ou substituições de um perfil do grafo. Os valores desta métrica são obtidos por *logs* do sistema que incrementam um contador a cada vez que o *buffer* do Cliente está vazio e este solicita um novo bloco. O *log* do sistema é computado no final da execução dos testes e, então, é feita a identificação de todas as interrupções constatadas.

Como pode ser observado há duas métricas a mais do que informado na seção 5.2.1. Estas métricas visam identificar se houve interrupção na reprodução do conteúdo, não são habitualmente consideradas em várias análises e foram incluídas para permitir identificar se as otimizações propostas afetam a QoE do Cliente. Sendo assim, elas são incluídas nessa tese como um critério adicional para avaliar o serviço de localidade quando usado em aplicações P2P que consomem conteúdo de vídeo.

Para todos os testes, foram realizadas, todas as medições, sendo cada teste descrito da seguinte forma:

- Apresentação do estado do sistema na topologia do ambiente de teste no momento inicial, indicando onde os conteúdos estão disponíveis e o uso do serviço de localidade;
- Definição do *peer* que irá solicitar o conteúdo; e
- Descrição do comportamento esperado do sistema relacionado à lista de *peers* retornada pelo *Tracker* e uso dos enlaces de rede.

O tipo de serviço usado nos testes foi o de VoD, sendo que o conteúdo distribuído é um arquivo de vídeo com o tamanho de 388,8 Mib e com 4,5 minutos de duração. O arquivo do vídeo foi submetido ao sistema para preparação do conteúdo, que consiste na geração dos metadados (arquivos torrent) e alocação dos mesmos no repositório de torrents que está localizado junto com o *Proxy*. A preparação do conteúdo para distribuição foi feita com blocos de dois segundos, resultando em 135 arquivos torrent (metadados).

6.1 Testes

A solução proposta nessa tese tem por objetivo prover serviço de distribuição de conteúdo através de um sistema P2P otimizado, denominado P2PM. Contudo, conforme identificado no Capítulo 2 (Figuras 2, 3 e 4), na atualidade a maior parte do conteúdo trafegado na Internet é do tipo multimídia (vídeo em sua maioria) e é distribuído usando serviços que não são P2P. O relatório técnico sobre tipo de tráfego Internet, apresentado por (SANDVINE, 2012), identifica que o conteúdo de vídeo é

tipicamente disponibilizado através de soluções cliente-servidor tipo *Content Distribution Network* (CDN). Neste sentido, serão incluídos também nos testes dessa tese um experimento que possui as propriedades de um sistema cliente-servidor tradicional para sistemas IPTV (LOHMAR; KRITZNER, 2008; VESTAL, 2012). A maioria das CDN usam o modelo cliente-servidor para disponibilizar o conteúdo em um ponto próximo aos Clientes através do emprego de servidores de *cache*. Esses servidores de *cache* normalmente consistem em cópias (redundância) do conteúdo que originalmente está disponível em um servidor remoto e que são acessados pelos Clientes através do redirecionamento da requisição da localização de um servidor principal para o servidor de *cache* (quando houver um disponível).

Foram definidos cinco cenários de teste para a avaliação do Sistema de IPTV com serviço de localidade. Os testes foram identificados da seguinte forma:

Teste + <modelo (**CS/P2P**, CS - Cliente Servidor e P2P - *Peer to Peer*)> + <**número** de identificação do teste>.

Descrição dos testes:

- **Teste CS 0:** O conteúdo é disponibilizado via *streaming* através do aplicativo VLC original no servidor *Proxy* sendo consumido por uma aplicação VLC em execução no *Peer1*. O objetivo é verificar o desempenho do serviço em um sistema cliente-servidor. Este teste também serve como base à comparação com os resultados dos demais testes e representa um ambiente básico de cliente-servidor.
- **Teste P2P 0:** Está baseado em apenas um *Tracker* BitTorrent. O conteúdo está disponível em todos os *peers* exceto pelo *Peer1* que solicitará o conteúdo. Este teste também serve como base à comparação com os resultados dos demais testes e representa o BitTorrent padrão.
- **Teste P2P 1:** Serviço de localidade habilitado com todos os *Trackers* P2PM.

O conteúdo está disponível em todos os *peers* exceto pelo *Peer1*. O objetivo é verificar o desempenho do serviço em uma situação com *peers* com o conteúdo solicitado disponível no mesmo domínio do *peer* requisitante.

- **Teste P2P 2:** Serviço de localidade habilitado com todos os *Trackers*. O conteúdo está disponível apenas nos *peers* do Domínio *Tracker1*. O *Peer5* (Domínio *Tracker2*) solicita um conteúdo não disponível no seu domínio. O objetivo é verificar o desempenho do sistema e a aplicação do serviço de localidade.
- **Teste P2P 3:** Serviço de localidade habilitado com todos os *Trackers*. O conteúdo está disponível em todos os *peers* exceto pelo *Peer1*. O *Peer1* solicita um conteúdo e durante a sua obtenção as condições de uso dos enlaces são alteradas. Este teste visa identificar os aspectos dinâmicos do serviço de localidade, através das listas de *peers* fornecidas pelo *Tracker1* à medida que as condições da rede mudam.
- **Teste P2P 4:** Serviço de localidade habilitado com todos os *Trackers*. O conteúdo está disponível em todos os *peers* exceto pelo *Peer1*. O *Peer1* solicita um conteúdo e durante a sua obtenção é atingida a condição para substituição do perfil do grafo no *Tracker1*. Este teste visa mostrar como o valor de *Prioridade* pode influenciar no peso dos enlaces e consequentemente, na lista de *peers* fornecida pelo *Tracker1* à medida que as condições da rede mudam.

Cada teste foi executado vinte vezes sequencialmente, sendo que ao final de cada sequência é calculada a média e o desvio padrão. No início de cada execução, o sistema é reinicializado para assegurar as mesmas condições de execução no computador (processador e memória) em todas as execuções. Cabe ainda ressaltar que, a rede onde os experimentos são executados está completamente isolada a fim de evitar que outros tipos de tráfegos possam comprometer a exatidão dos testes.

6.1.1 Teste CS 0 - Cliente-servidor

O Teste CS 0 representa um sistema cliente-servidor tradicional onde o conteúdo é transmitido de um computador diretamente para outro através de algum protocolo. Entretanto, esse mesmo cenário pode ser aplicado também para o caso de uma *Content Distribution Network* (CDN) que não possui um servidor *cache* próximo ao Cliente. Uma CDN busca disponibilizar cópia do conteúdo próximo ao solicitante através de servidores de *cache*, porém quando não há um servidor *cache* é necessário obter o conteúdo de uma localização mais remota.

A transmissão do conteúdo de vídeo foi realizada através do serviço nativo de *streaming* da aplicação VLC empregando o protocolo *Real Time Protocol* (RTP) / MPEG. A aplicação VLC possui um mecanismo de controle de fluxo interno que evita o uso de mais largura de banda do que o necessário para reproduzir o conteúdo de vídeo e, ao mesmo tempo, atrasa a apresentação para armazenar (*buffer*) o conteúdo se não houver largura de banda suficiente (LATTRE et al., 2011). Esse tipo de mecanismo é comumente empregado por CDN (POESE et al., 2012; LIU et al., 2012). Inicialmente é natural imaginar que obter o conteúdo o mais rapidamente possível seja o mais adequado. Contudo, tratando-se de conteúdo de vídeo que tem requisitos de tempo críticos como atraso e variação no atraso na apresentação dos quadros de vídeo, é possível que nem sempre o fato do usuário começar a assistir um conteúdo signifique que ele irá assisti-lo por completo. Nessa situação, se todo o conteúdo for obtido o mais rapidamente possível poderá haver o uso desnecessário dos recursos da infraestrutura, se o Cliente deixar de reproduzir o conteúdo antes do final (desiste de assistir o conteúdo, e.g., desinteresse no conteúdo). Além disso, Cliente pode desistir de assistir o vídeo se perceber a existência de muito atraso e não regularidade na apresentação; o armazenamento do vídeo antes visa evitar este problema. Tais fatores levam a implementação de mecanismos de controle fluxo, empregados no VLC.

A Figura 22 mostra a topologia do *testbed* e indica o lugar onde o conteúdo está

disponível (caixa amarela), sendo que o Cliente localizado no *Peer1* irá solicitar este conteúdo.

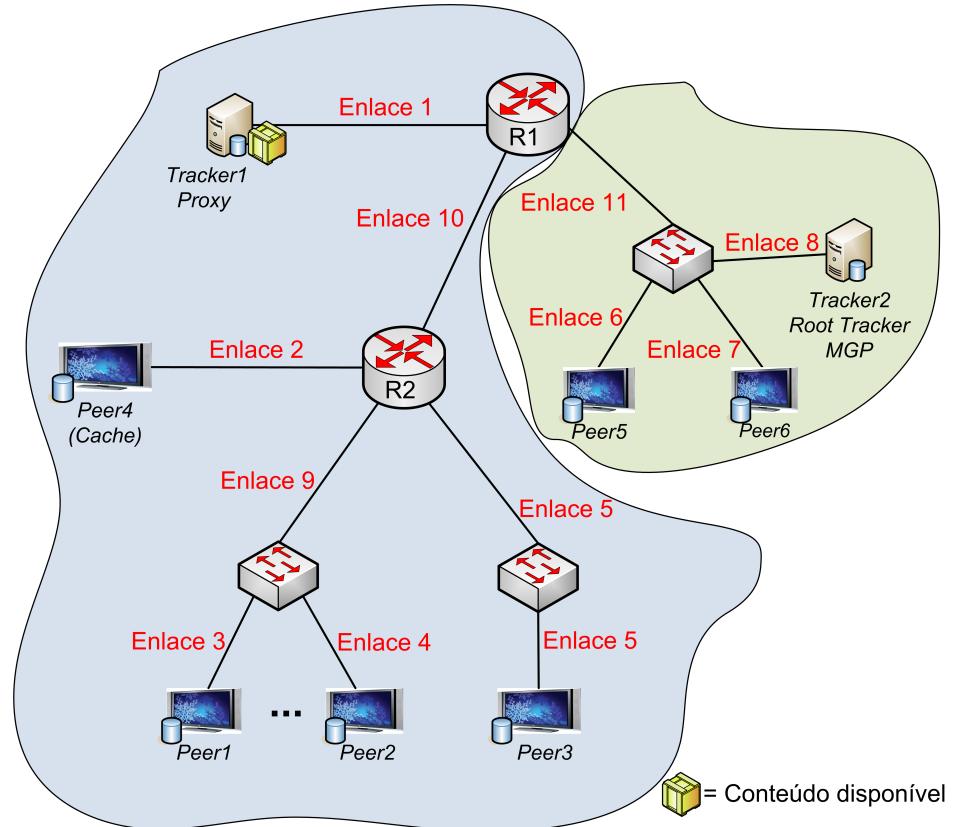


Figura 22: Teste CS 0: Disposição dos elementos do sistema e topologia da rede

A Figura 23 apresenta a consolidação do montante de dados trafegados nos enlaces de rede nos testes realizados.

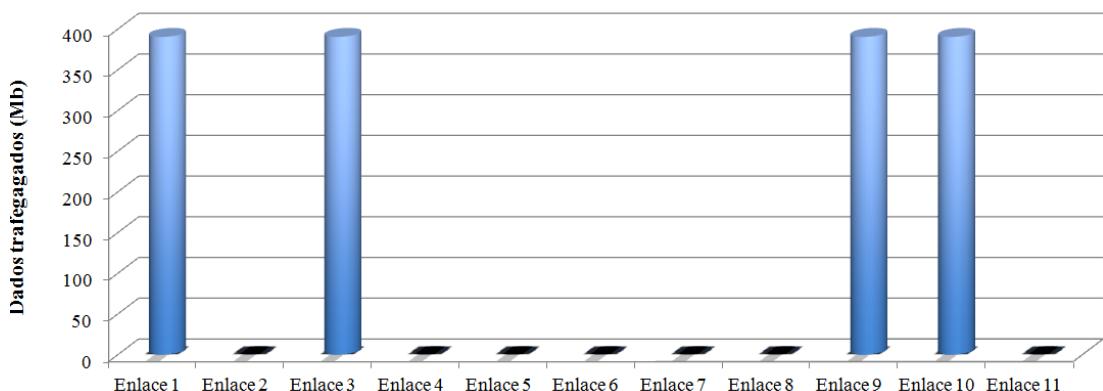


Figura 23: Teste CS 0 - Quantidade de dados trafegados nos enlaces

A Figura 23 mostra o resultado esperado do uso dos enlaces (Enlaces: 1, 3, 9 e 10), visto que o conteúdo é obtido de uma única fonte.

Neste teste, a medição da latência para inicialização da apresentação do conteúdo foi de 2213ms, não ocorrendo qualquer interrupção na reprodução do conteúdo no Cliente. Contudo, o valor acima dos dois segundos chama a atenção. Analisando a vazão mensurada no experimento (Figura 24) e o comportamento do VLC, foi identificado que há no início do *streaming* um período de armazenamento (*buffer*) que é típico do VLC. Sendo assim, é compreensível o valor da latência inicial identificado nos testes.

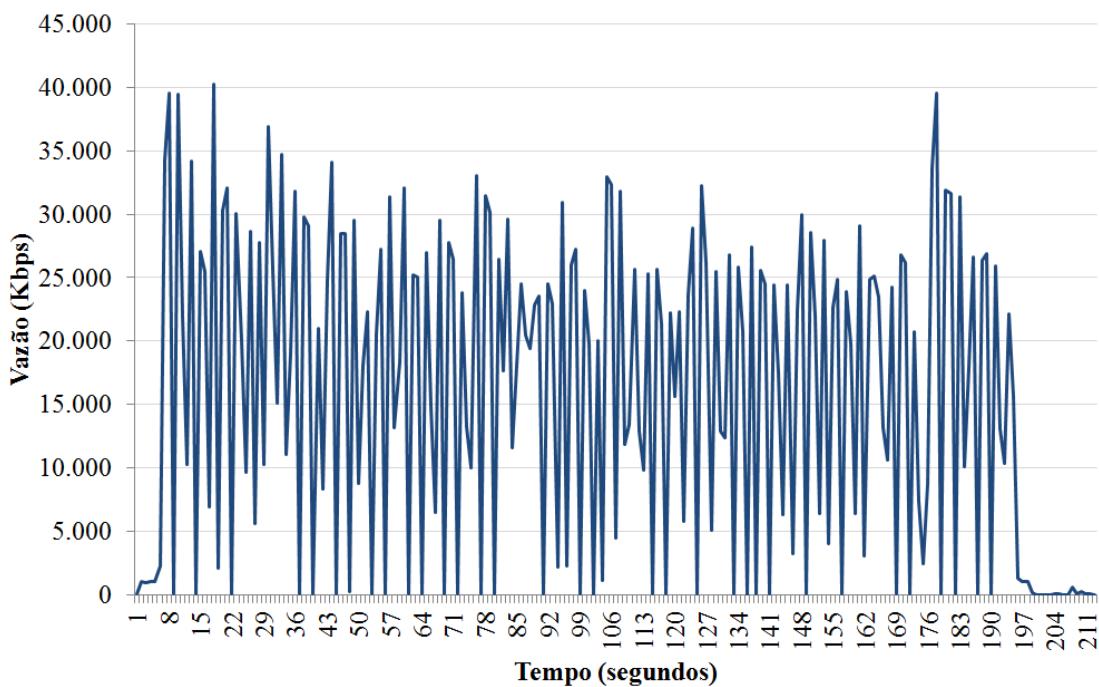


Figura 24: Vazão durante a obtenção do conteúdo no Teste CS 0.

6.1.2 Teste P2P 0 - BitTorrent padrão

O Teste P2P 0 representa um sistema BitTorrent tradicional, no qual usualmente há apenas um *tracker* que coordena todos os *peers* no *swarm*. A Figura 25 mostra a topologia do ambiente de teste e indica em quais lugares o conteúdo está disponível, sendo que o *Peer1* é o único *peer* sem o conteúdo.

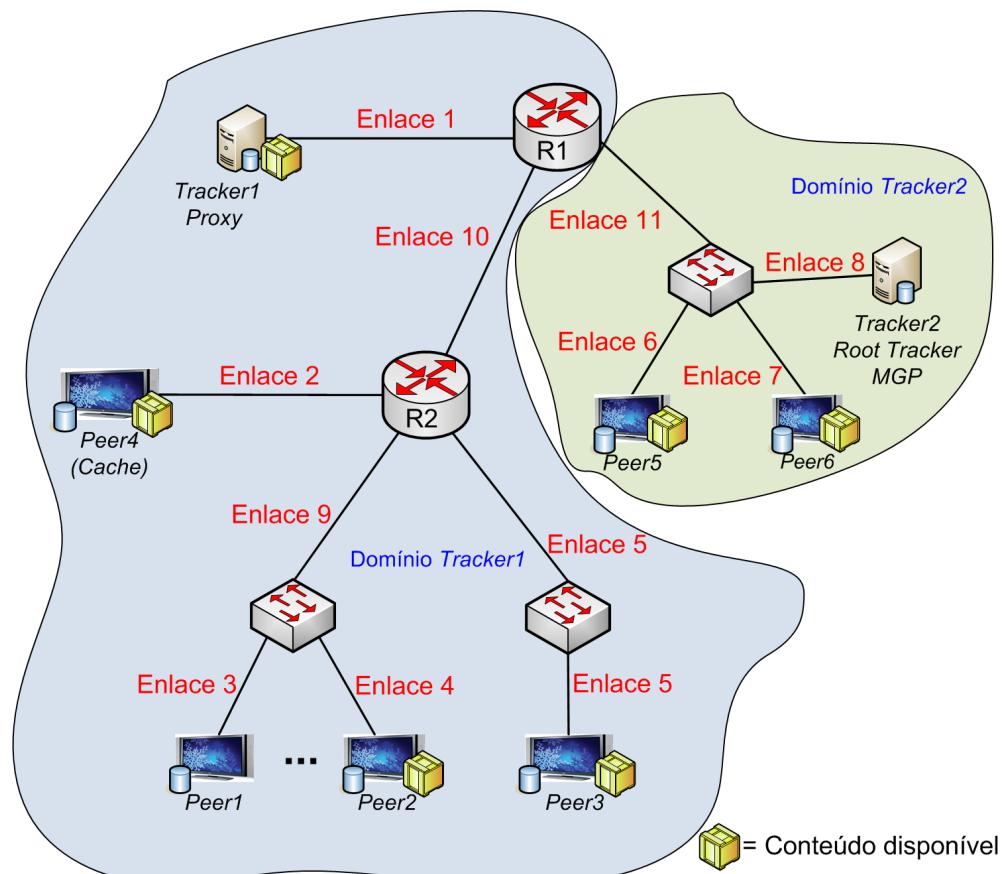


Figura 25: Teste P2P 0: Disposição dos elementos do sistema IPTV e topologia da rede

A Figura 26 apresenta a consolidação do montante de dados trafegados nos enlaces de rede. Observando o gráfico é perceptível a ausência de tráfego no Enlace 8, o que é um resultado esperado pelo fato de não haver conteúdo disponível neste local (esse computador contém apenas módulos que são usados pelo serviço de localidade).

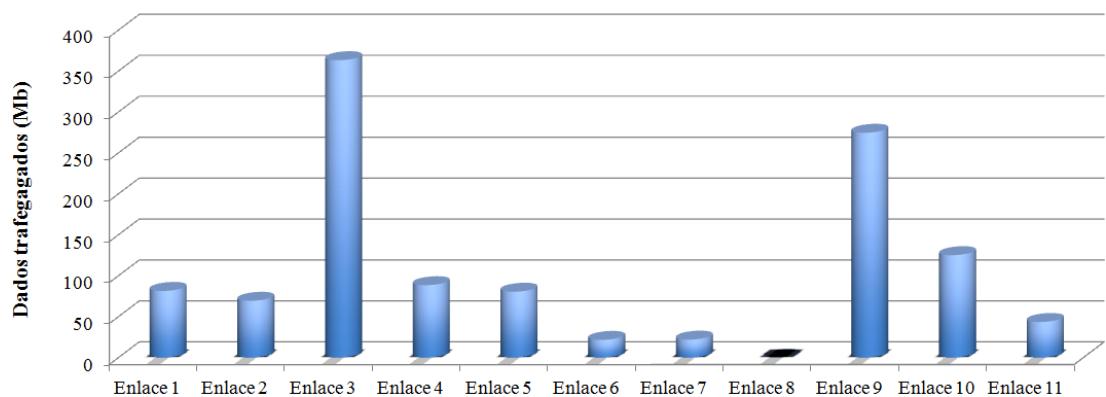


Figura 26: Teste P2P 0 - Quantidade de dados trafegados nos enlaces

Analisando o gráfico (Figura 26) é possível constatar a existência de tráfego em todos os enlaces (com exceção do Enlace 8). Isto é explicado pelo comportamento do protocolo BitTorrent quando um *peer (leecher)* entra em um *swarm*, que é de retornar um lista de *peers* com 50 *peers* que também têm interesse no conteúdo. Deste modo, o *Peer1* recebeu uma lista com todos os *peers* do sistema.

A Figura 27 permite identificar uma transferência de dados mais intensa dos *peers* mais próximos. Esse comportamento é resultante da proximidade, em termos de saltos de roteamento, entre o *peer* solicitante e os demais *peers* com conteúdo.

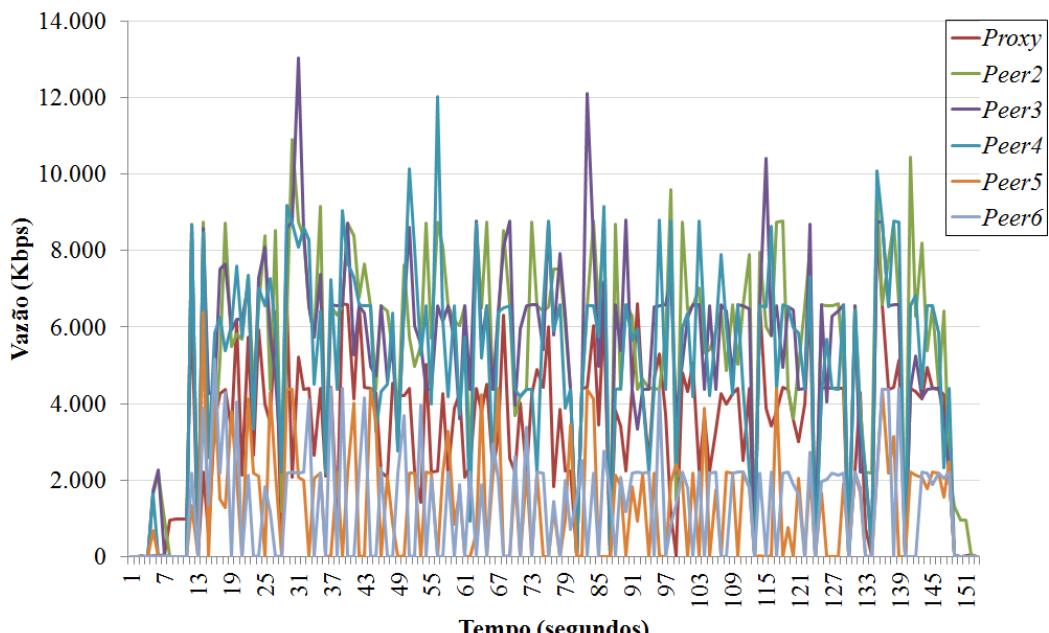


Figura 27: Teste P2P 0 - Vazão por *peers*

Outro comportamento do BitTorrent é o de buscar a obtenção do conteúdo do máximo de *peers* possíveis simultaneamente, fato que levou o *Peer1* a obter o conteúdo de todos os outros *peers* do sistema e que resultou no uso de vários enlaces.

Neste teste a medição da latência para inicialização da apresentação do conteúdo foi de 2054ms, não ocorrendo interrupção na reprodução do conteúdo no Cliente. A aferição do tempo médio para a obtenção dos blocos obteve o valor de 1095ms, tempo abaixo de dois segundos que gera como consequência um acúmulo de blocos na área de armazenamento (*buffer*) do Cliente e reduz a probabilidade de interrupção na reprodução do conteúdo.

6.1.3 Teste P2P 1 - Conteúdo disponível no domínio da solicitação

O Teste P2P 1 apresenta o mesmo cenário inicial (Figura 28) da disposição dos conteúdos do Teste P2P 0. Neste cenário o serviço de localidade encontra-se habilitado e o *Peer1* solicita um conteúdo que está disponível nos outros *peers* do seu domínio. É importante destacar que no momento da solicitação do *Peer1*, todos os grafos do sistema estão atualizados e as tabelas de conteúdo já possuem o conteúdo registrado.

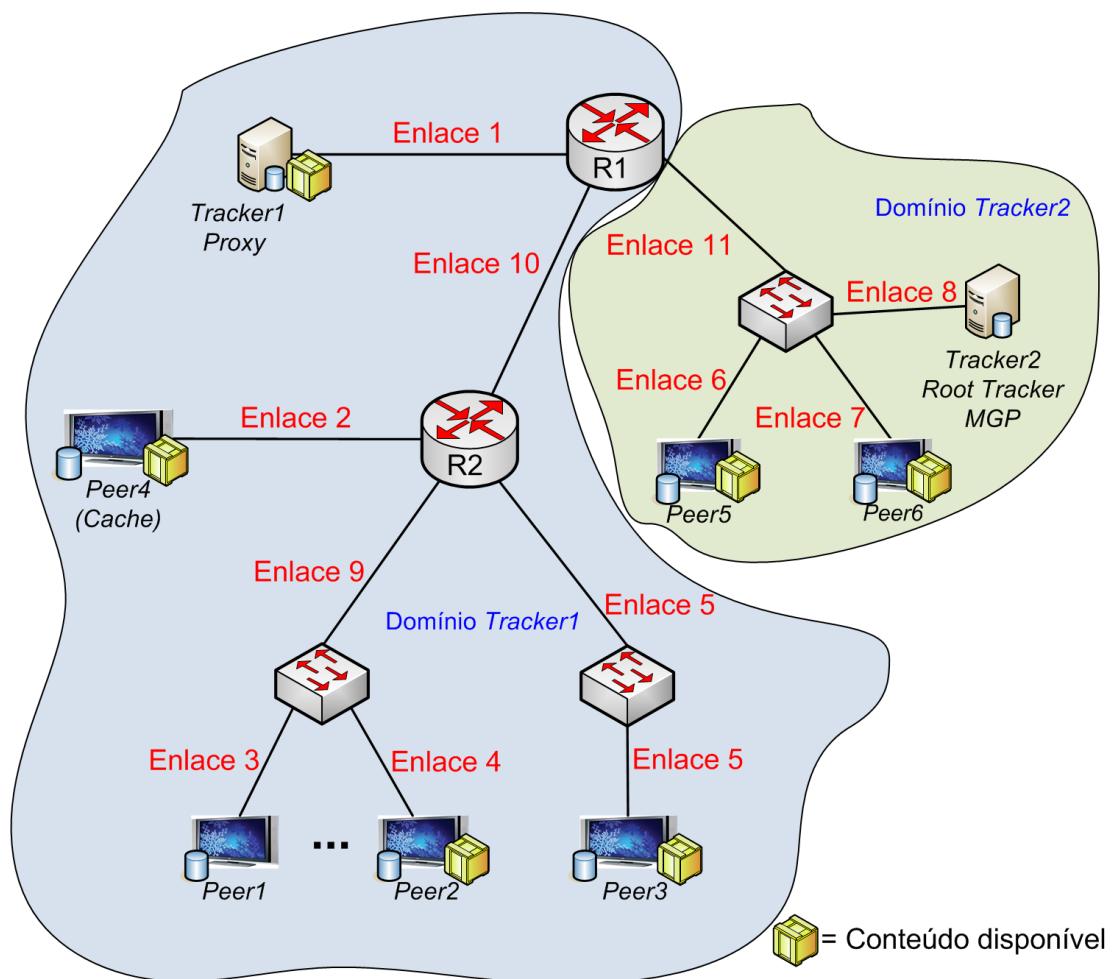


Figura 28: Teste P2P 1 - Disposição dos elementos do sistema IPTV e topologia da rede

A Figura 29 apresenta a consolidação do montante de dados trafegados nos enlaces de rede. Neste gráfico é possível observar o uso de poucos enlaces de rede, porém os enlaces utilizados possuem um montante maior de dados trafegados.

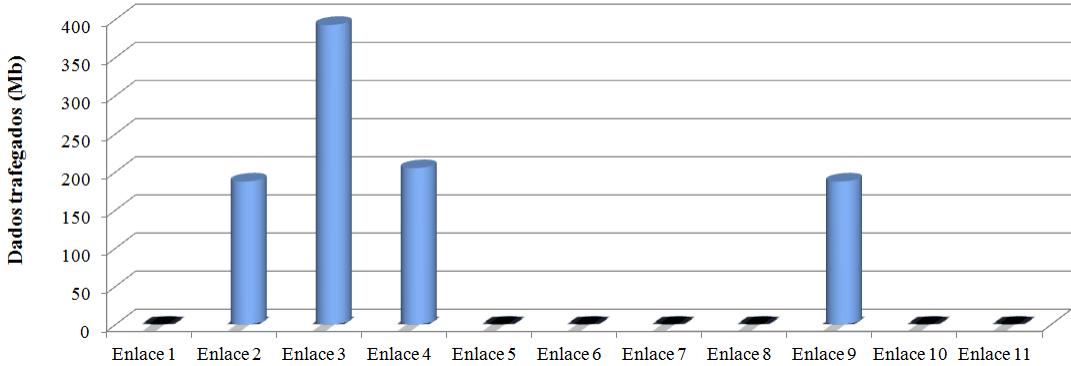


Figura 29: Teste P2P 1 - Quantidade de dados trafegados nos enlaces

O uso do serviço de localidade, fornecido pela arquitetura de *trackers* hierárquicos, implicou no *Peer1* solicitar o conteúdo ao *Tracker1*. O *Tracker1* computou o melhor caminho entre os *peers* com o conteúdo (*Peer2*, *Peer3*, *Peer4* e *Proxy*) e o *Peer1* através do cálculo do peso de cada caminho usando o Algoritmo de Dijkstra. Como resultado obteve os melhores *peers* (*Peer2* e *Peer4*) que têm capacidade de atender a solicitação. Sendo assim, é perceptível o tráfego de dados apenas nos enlaces que compreendem o caminho entre *Peer1* e *Peer2* e entre *Peer1* e *Peer4* (Figura 30).

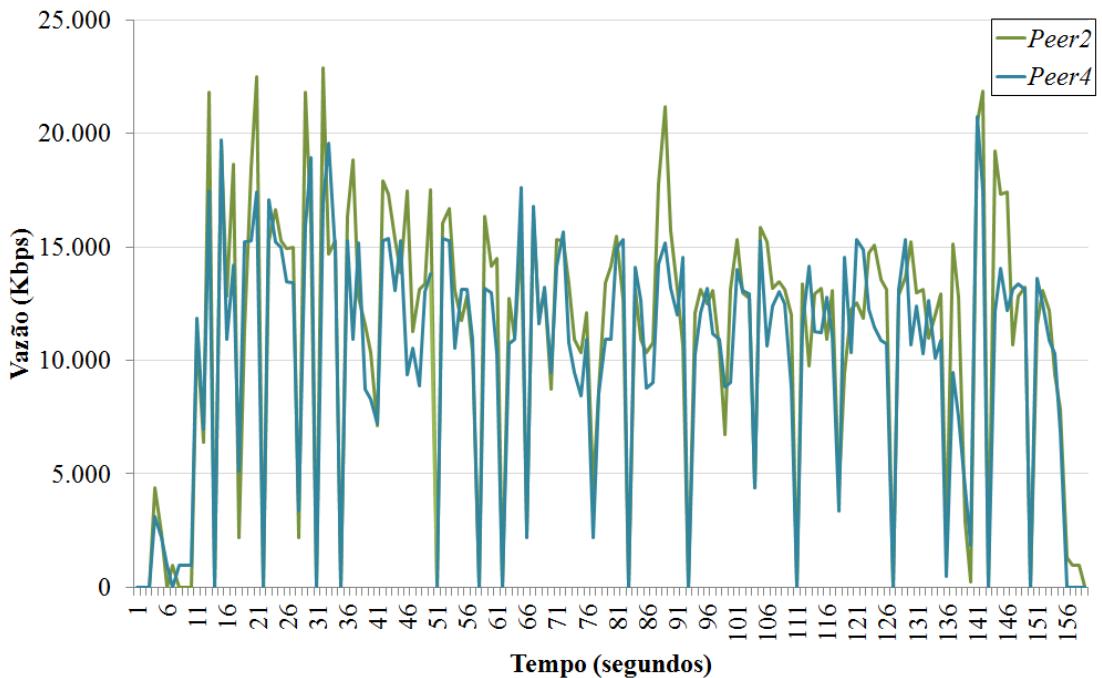


Figura 30: Teste P2P 1 - Vazão por *peers*

Foi identificado um tráfego de controle pequeno, total de 340Kb, no sentido do *Peer1* para o *Tracker1* (Figura 31). Esse tráfego de controle é composto de notificações

para atualização da Tabela de Conteúdo (248Kb) e obtenção dos torrents (92Kb) que estão armazenados no *Proxy*. Não foi identificado tráfego de controle entre *Tracker1* e *Root Tracker* devido ao conteúdo já estar disponível no Domínio *Tracker1*.

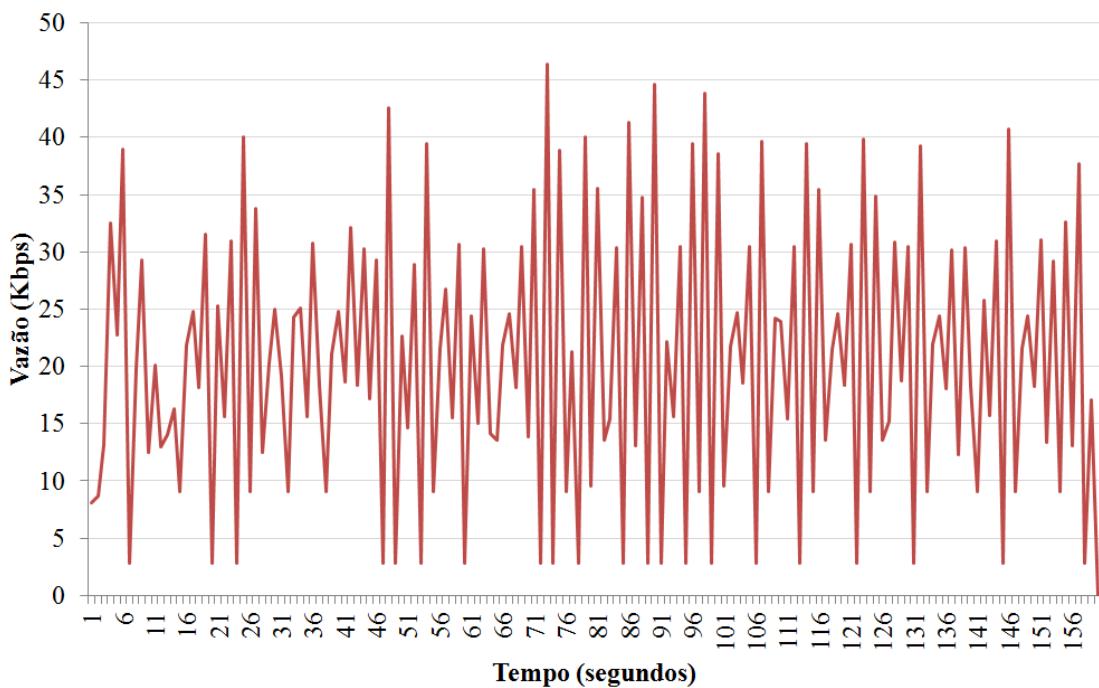


Figura 31: Teste P2P 1 - Tráfego de controle entre *Peer1* e *Tracker1*

Neste teste o valor médio na medida da latência para inicialização da apresentação do conteúdo foi de 1681ms e o tempo médio para a obtenção de cada bloco foi de 1191ms, não havendo interrupção do conteúdo em qualquer dos testes executados.

6.1.4 Teste P2P 2 - Conteúdo indisponível no domínio da solicitação

O Teste P2P 2 apresenta o cenário inicial (Figura 32) com uma alocação dos conteúdos elaborada para provocar a obtenção do mesmo em outro domínio e, assim, analisar o comportamento da arquitetura de *trackers* hierárquicos. Neste cenário o serviço de localidade encontra-se habilitado e o *Peer5* solicita um conteúdo que não está disponível nos *peers* do seu domínio (*Tracker2*). É importante destacar que no momento da solicitação do *Peer5*, todos os grafos do sistema estão atualizados e as tabelas de conteúdo já possuem o conteúdo registrado.

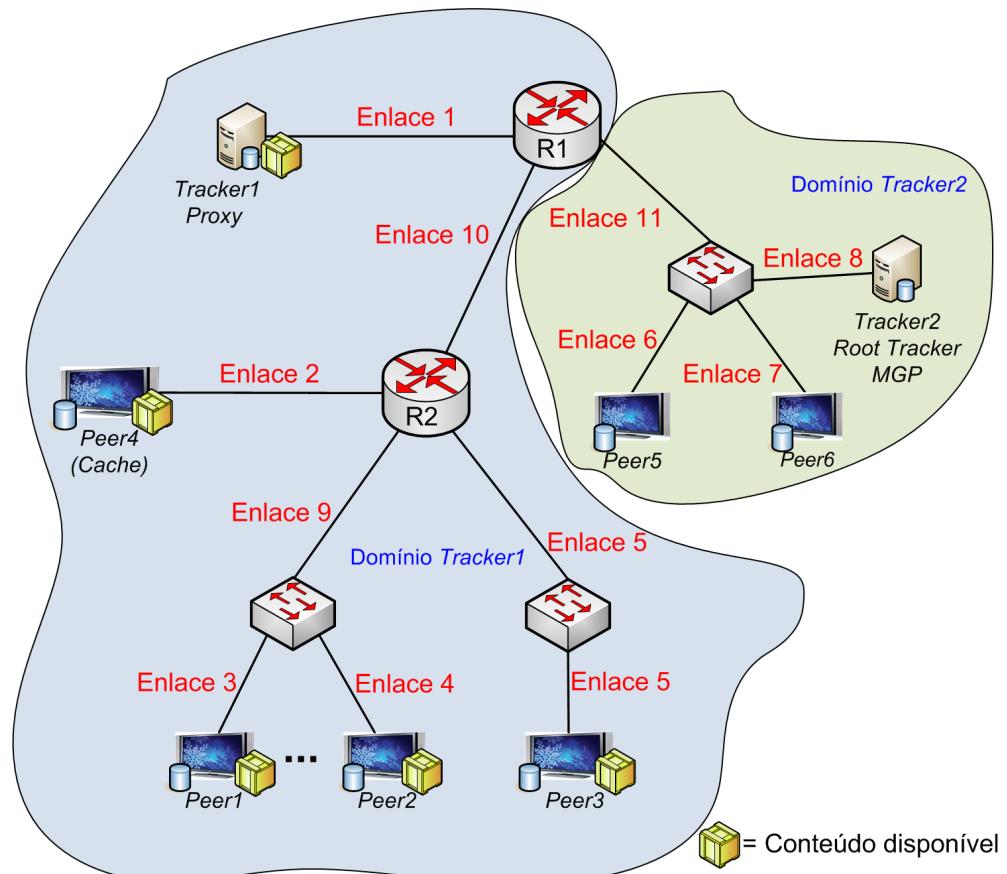


Figura 32: Teste P2P 2 - Disposição dos elementos do sistema IPTV e topologia da rede

A Figura 33 apresenta a consolidação do montante de dados trafegados nos enlaces de rede.

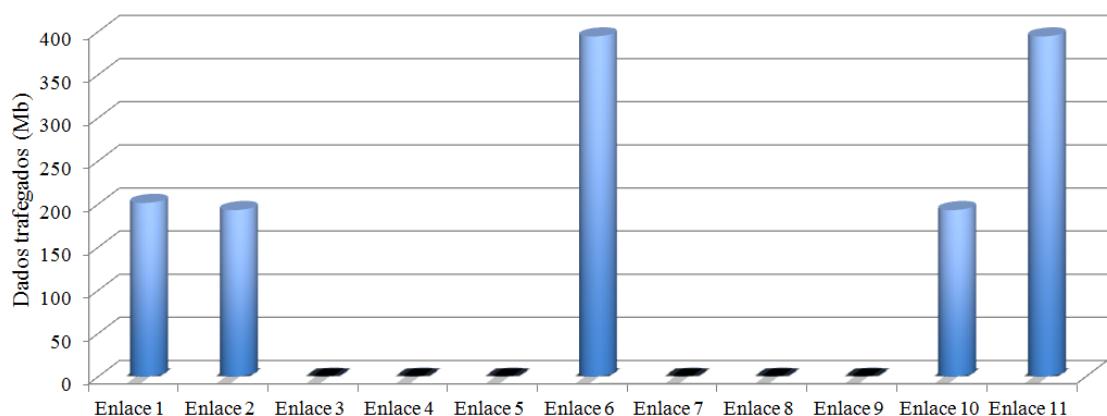


Figura 33: Teste P2P 2 - Quantidade de dados trafegados nos enlaces

Quando o *Tracker2* detecta que o conteúdo solicitado pelo *Peer5* não está disponível no seu domínio, ele contacta o *Root Tracker* para obter a lista de *trackers* onde

o conteúdo está disponível. O *Root Tracker* então usa o seu grafo para avaliar quais os *trackers* mais próximos e elabora uma lista, sendo que no cenário apresentado só existe o *Tracker1* e ele possui o conteúdo. Aqui é importante ressaltar que o *Root Tracker* também computa o melhor caminho entre o *tracker* que fez a solicitação e os *trackers* que possuem o conteúdo. Em seguida, o *Tracker2* contacta o *Tracker1* e solicita uma lista de *peers*. Neste contexto, o *Tracker1* não possui o *peer* solicitante no seu domínio, então ele usa a conexão de saída do Domínio *Tracker1* como referência. Consequentemente, a lista de *peers* retornada pelo *Tracker1* contém o *Proxy* e *Peer4*. Por fim, o *Tracker2* retorna a lista de *peers* obtida para o *Peer5* que obtém o conteúdo do Domínio do *Tracker1* usando os Enlaces 1, 2, 6, 10 e 11. A Figura 34 apresenta um histórico da vazão durante um teste.

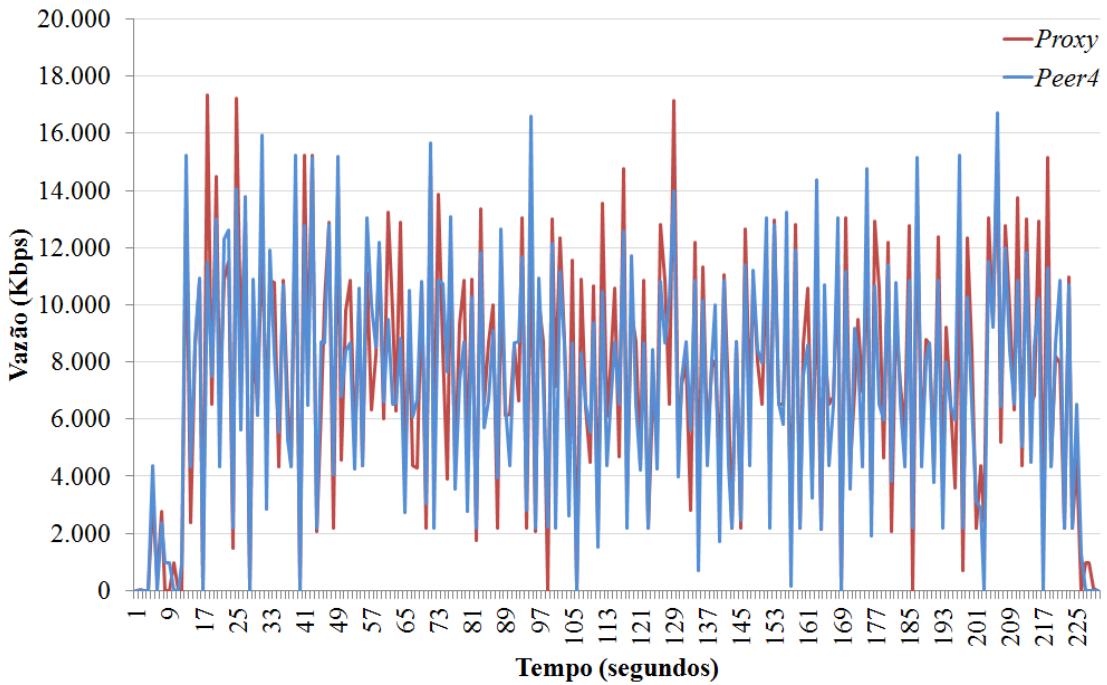


Figura 34: Teste P2P 2 - Vazão por *peers*

Neste teste também foi identificado um pequeno montante (total de 320Kb) de dados trafegado entre o *Peer5* e o *Tracker2* referente a notificações de conteúdo obtido para a atualização da Tabela de Conteúdo (TC). As comunicações entre o *Root Tracker* e *Tracker2* com o *Tracker1* contabilizaram um total de 4.960Kb. A latência para inicialização da apresentação do conteúdo foi de 2309ms, com valor médio para a obtenção

dos blocos de 1703ms. Foi detectada uma interrupção da apresentação do conteúdo em oito das dez execuções do teste, sendo que as duas execuções que não indicaram interrupções tiveram a média registrada de 1671ms e 1699ms. Estas interrupções são analisadas na seção 6.2.1.

6.1.5 Teste P2P 3 - Mudança nas condições da rede

O Teste P2P 3 (Figura 35) apresenta o mesmo cenário inicial da disposição dos conteúdos dos Testes P2P 0 e 1. Neste cenário o serviço de localidade encontra-se habilitado e o *Peer1* solicita um conteúdo que está disponível nos outros *peers* do seu domínio. Durante a obtenção do conteúdo as condições da rede são propositalmente alteradas para verificar a resposta do serviço de localidade.

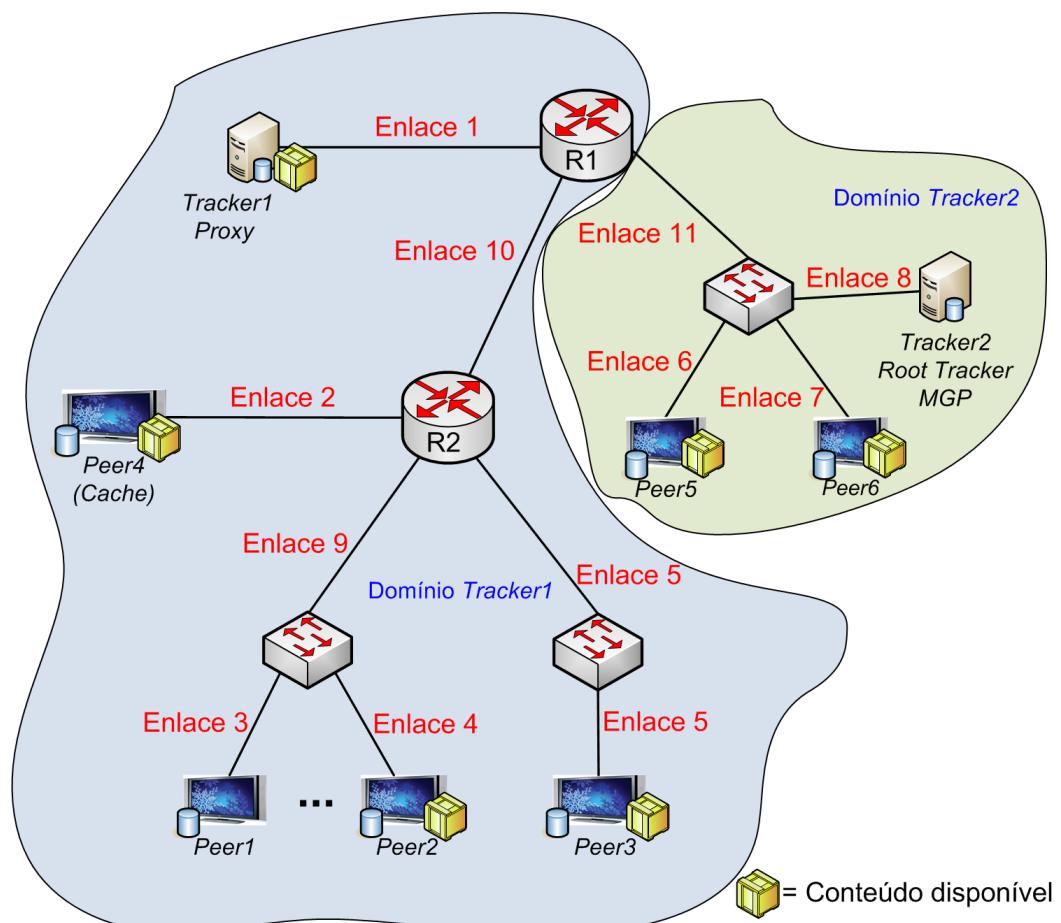


Figura 35: Teste P2P 3 - Disposição dos elementos do sistema IPTV e topologia da rede

Este teste inicia-se de forma idêntica ao Teste P2P 1, com *Peer1* obtendo o conteúdo do *Peer2* e *Peer4*. Então, 30 segundos após o início da obtenção do conteúdo, é injetado na rede um tráfego de 1.2 Gb oriundo da cópia de um arquivo entre os computadores onde estão localizados o *Peer4* e *R2*. O tráfego resultante afeta negativamente o Enlace 2, implicando em uma mudança do peso dessa aresta no grafo do *Tracker1*. Esta mudança de peso tem como consequência que a lista de *peers* para obter os blocos seguintes, retornadas pelo *Tracker1* ao *Peer1*, contém *Peer2* e *Peer3*. A Figura 36 apresenta a consolidação do volume de dados trafegados nos enlaces de rede.

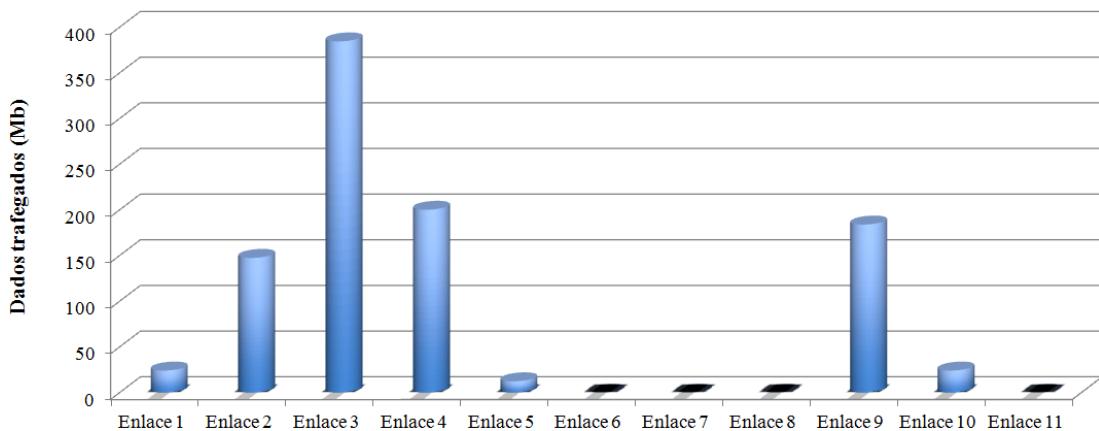


Figura 36: Teste P2P 3 - Quantidade de dados trafegados nos enlaces

Analizando o grafo e a topologia da rede é identificado que tanto o *Peer3* como o *Proxy* encontram-se equidistantes, em termos de saltos, do *Peer1*. Porém, entre o *Peer1* e *Proxy* há um pequeno tráfego de controle em função do *Tracker1* estar no mesmo computador do *Proxy*. Sendo assim, o grafo reflete uma pequena diferença entre essas duas opções e escolhe o *Peer3* por ser o mais favorável. Uma vez que o *Peer3* inicia o fornecimento de conteúdo para o *Peer1*, as condições dos enlaces entre estes *peers* se alteram em função do tráfego associado. Deste modo, na próxima atualização do grafo que ocorrer o *Peer3* não é mais vantajoso do que o *Proxy*, sendo esse incluído nas próximas listas de *peers* retornadas do *Tracker1* ao *Peer1*. Após o término da cópia do arquivo, o Enlace 2 tem seu peso novamente atualizado, sendo então o *Peer4* novamente preferido pelo *Tracker1* para fornecer o conteúdo ao *Peer1*. A Figura 37 apresenta o histórico do tráfego gerado por todos os *peers* envolvidos no

teste, enquanto a Figura 38 destaca o tráfego do *Peer3*, *Peer4* e *Proxy* para melhor identificar as mudanças nas listas de *peers*.

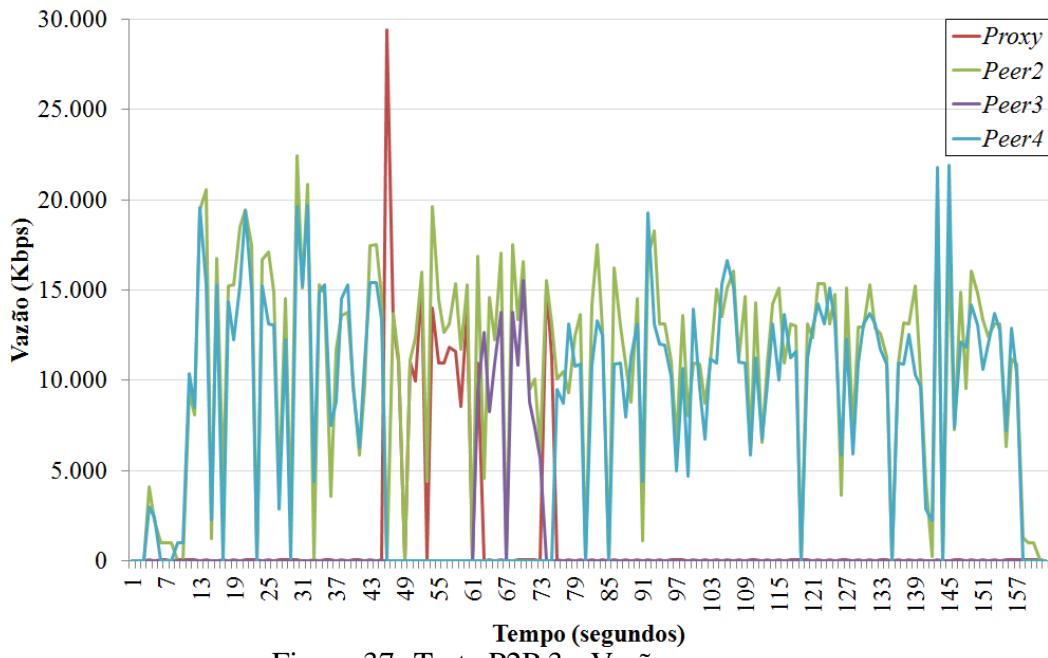


Figura 37: Teste P2P 3 - Vazão por peers

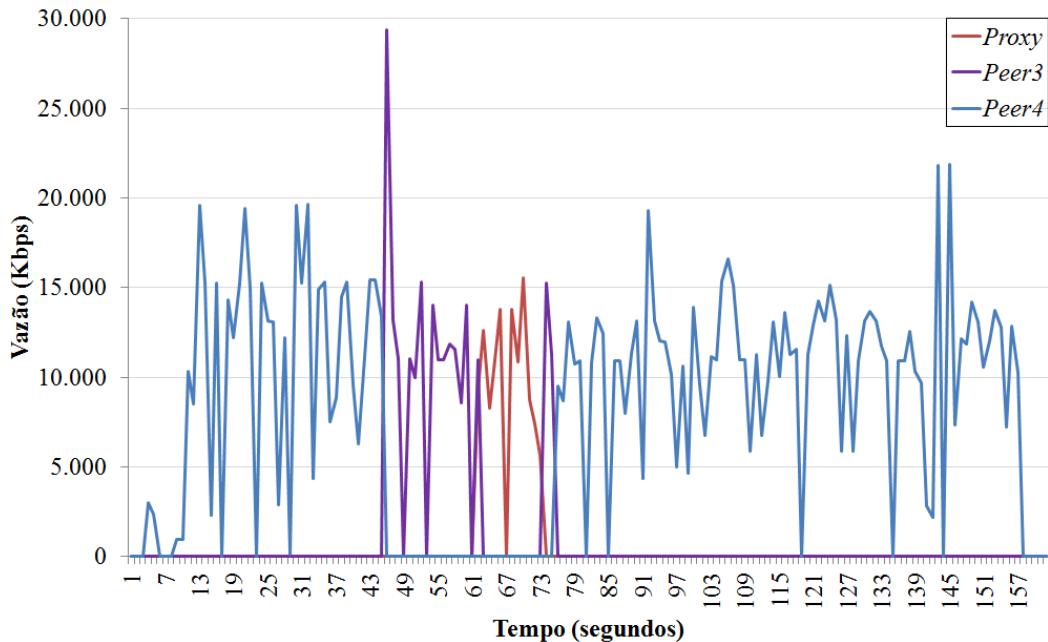


Figura 38: Teste P2P 3 - Alternância entre uso de *peers* em função das condições da rede

Neste teste também foi identificado (similar ao Teste P2P 1) um tráfego de controle de 340Kb no sentido do *Peer1* para o *Tracker1*, que ocorreu devido a notificações para atualização da Tabela de Conteúdo (TC) e obtenção dos torrents que estão armazenados no *Proxy*.

O valor médio da latência para inicialização da apresentação do conteúdo foi de 1708ms, sendo aferido um tempo médio para a obtenção de cada bloco no valor de 1196ms. Neste cenário não foi detectada a interrupção do conteúdo em nenhuma das execuções do teste.

6.1.6 Teste P2P 4 - Substituição do perfil do grafo no *Tracker1*

O Teste P2P 4 (Figura 39) apresenta o mesmo cenário inicial da disposição dos conteúdos dos Testes P2P: 0, 1 e 3. Neste cenário o serviço de localidade encontra-se habilitado e o *Peer1* solicita um conteúdo que está disponível nos outros *peers* do seu domínio, sendo que durante a obtenção do conteúdo é atingida uma condição para substituição do perfil do grafo no *Tracker1*.

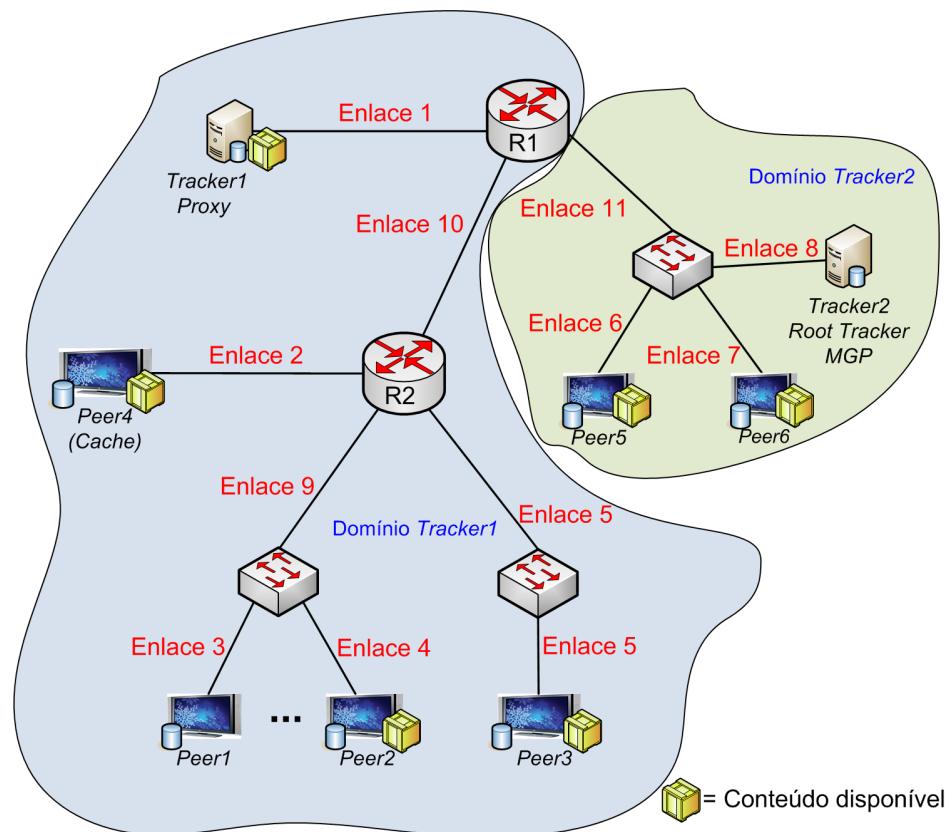


Figura 39: Teste P2P 4 - Disposição dos elementos do sistema IPTV e topologia da rede

Neste experimento foi definida uma condição de tempo para substituição do grafo no *Tracker1*, que quando atingida substitui o perfil do grafo ativo por outro. Essa condição foi configurada através do MGP (Módulo Gerenciador de Perfis) com objetivo de alterar os valores do critério *Prioridade* de modo a demonstrar como ele pode ser usado para influenciar o *Tracker1* a retornar uma lista de *peers* diferente do que entregaria usando como critério apenas as condições de uso da rede. No novo perfil do grafo (*Tracker1*) foi definido o valor de *Prioridade* do Enlace 2 como 100 (o valor padrão é 80). A Figura 40 apresenta a consolidação do montante de dados trafegados nos enlaces de rede.

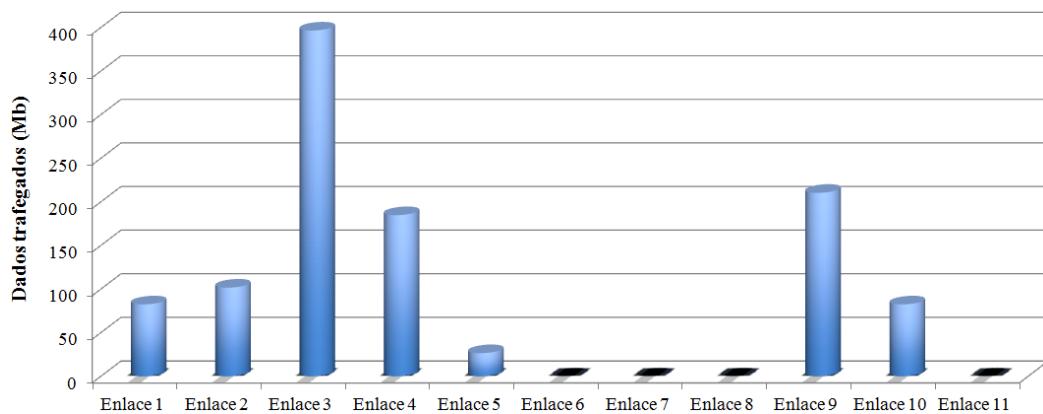
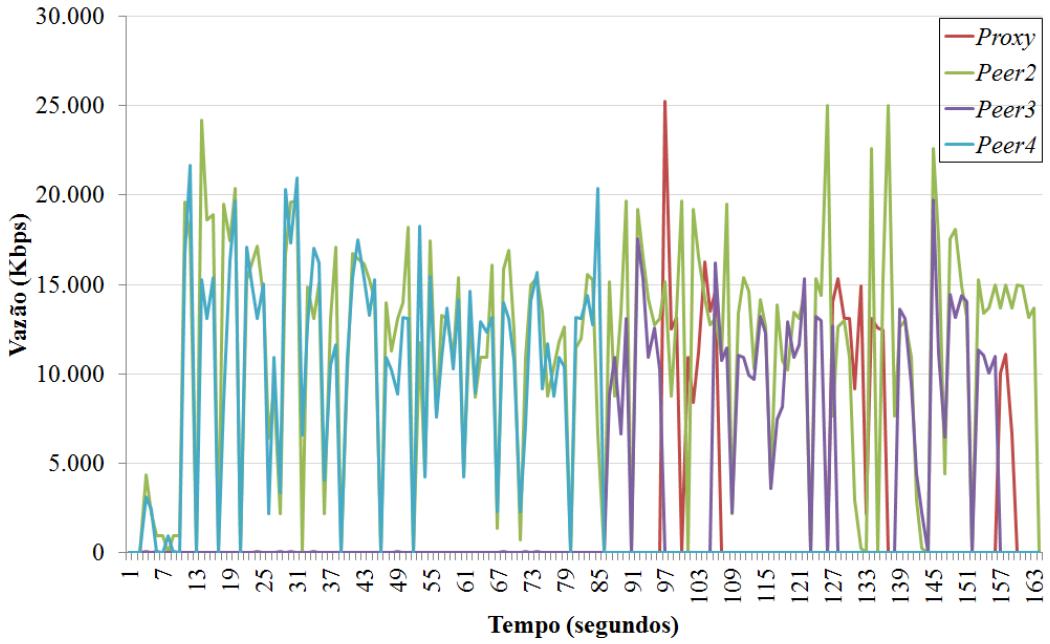
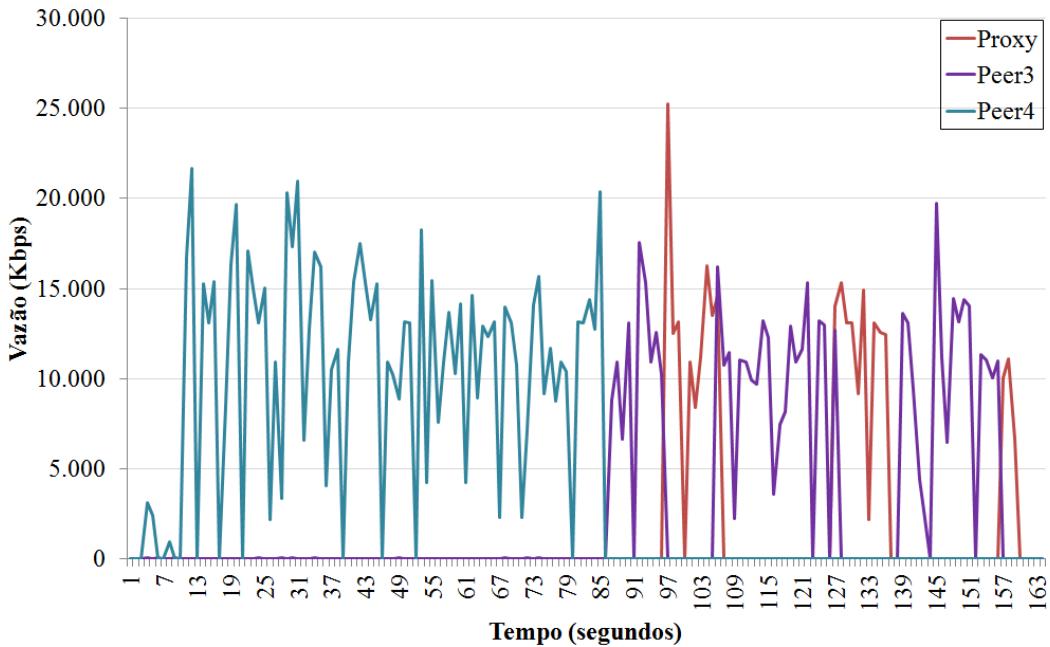


Figura 40: Teste P2P 4 - Substituição do perfil do grafo no *Tracker1*

A Figura 40 já identifica a utilização de enlaces que não são relacionados aos *Peer2* e *Peer4* (Enlaces 1 e 5), indicando o uso de outros *peers*. Analisando o teste foi verificado que este é iniciado de forma idêntica aos Testes P2P 1 e 3, com *Peer1* obtendo o conteúdo do *Peer2* e *Peer4*. Então, 80 segundos após o início da obtenção de conteúdo, o *Tracker1* atinge a condição para mudança do perfil do seu grafo. Assim que termina substituição do perfil, o *Tracker1* altera a lista de *peers* retornando agora o *Peer2* e *Peer3*. A Figura 41 apresenta o histórico da vazão de todos os *peers* envolvidos neste teste, enquanto a Figura 42 permite observar em específico o comportamento do tráfego dos *Peer3*, *Peer4* e *Proxy*.

Figura 41: Teste P2P 4 - Vazão por *peers*Figura 42: Teste P2P 4 - Alternância entre uso de *peers* em função das mudança do perfil do grafo do *Tracker1*

Analizando o perfil do grafo (Apêndice B.2 e a Figura 39) é constatado que tanto o *Proxy* como o *Peer3* possuem a mesma “distância” do *Peer1*. Então, na próxima atualização de valores do grafo, o *Tracker1* irá identificar que já existe tráfego no *Proxy* (tráfego de controle entre *Peer1* e *Tracker1*) e identifica o caminho para o *Peer3* sendo

a melhor opção. Deste modo, a próxima lista de *peers* retornada contém o *Peer2* e *Peer3*. Similarmente ao que aconteceu no Teste P2P 2, haverá uma alternância na lista de *peers* retornada pelo *Tracker1* entre o *Peer3* e *Proxy* em função das condições da rede. Contudo, é possível observar (Figuras 41 e 42) que essa alternância é mais frequente em função do tempo de atualização dos dados do grafo (periodicamente a cada 15 segundos). Esta alternância continua acontecendo toda vez que o grafo é atualizado e ocorre até o final da obtenção do conteúdo. Entretanto é possível observar que em alguns casos o período antes da mudança de *peer* é maior do que 15 segundo. Este fato é resultante do intervalo de tempo para atualização do grafo, i.e., o *Peer3* é informado como melhor opção apenas por alguns segundos logo após o grafo ser atualizado, enquanto o *Proxy* permanece o intervalo inteiro até a próxima atualização como a melhor opção.

O valor médio da latência para inicialização da apresentação do conteúdo foi de 1749ms, com tempo médio para a obtenção dos blocos de 1164ms. Não foi detectada nenhuma interrupção do conteúdo durante todas as execuções.

6.2 Análise dos resultados

Os cenários de testes apresentados permitem identificar a versatilidade do uso do serviço de localidade proposto através da análise de suas métricas relacionadas e das informações coletadas. Na seção 6.1, cada cenário é testado e os resultados mensurados são avaliados independentemente sem considerar os resultados dos outros testes. Uma vez que o Teste P2P 0 representa um sistema apenas usando BitTorrent, este deve ser comparado com os Testes P2P: 1 e 2 para que sejam identificadas as diferenças referentes ao uso dos enlaces e no desempenho do sistema quando o serviço de localidade é utilizado. Por outro lado, também é interessante incluir na análise os resultados dos testes de cliente-servidor (Teste CS 0), para realizar uma comparação com os resultados da solução baseada em P2P.

6.2.1 Análise do tempo de inicialização, tempo médio para a obtenção dos blocos e interrupção da apresentação

A Tabela 9 lista os resultados obtidos em todos os testes para as métricas de tempo de inicialização, tempo médio para a obtenção dos blocos e interrupção da apresentação.

Tabela 9: Comparaçāo dos resultados dos testes: tempo de inicialização, tempo médio para a obtenção dos blocos e interrupção da apresentação

Teste	Tempo de Inicialização (ms)	Tempo Médio para a Obtenção do Bloco (ms)	Interrupção da Apresentação
Teste CS 0	2213	n.a.	Não
Teste P2P 0	2054	1095	Não
Teste P2P 1	1681	1191	Não
Teste P2P 2	2309	1703	Sim
Teste P2P 3	1708	1196	Não
Teste P2P 4	1708	1164	Não

A Tabela 9 mostra que a Latência de Inicialização do sistema hierárquico de *trackers*, com conteúdo local (Teste P2P 1), é o menor de todos os testes. Por outro lado, este mesmo critério apresentou no Teste P2P 2 a maior latência dos testes e teve como motivo as mensagens iniciais para obtenção da lista de *peers* de outro domínio que foi realizada entre *Tracker2*, *Root Tracker* e *Tracker1*. Contudo, todos os testes P2P apresentaram tempo bem inferior ao limite de cinco segundos que é aceito para sistemas P2P fornecendo serviço de vídeo/IPTV (MANZATO; FONSECA, 2011) e compatível com o limite aceito para TV a cabo de dois segundos (GREENGRASS; EVANS; BEGEN, 2009). Logo, a arquitetura de *trackers* hierárquicos demonstrou ser eficiente quanto à Latência de Inicialização.

Comparando os Testes P2P com os resultados do teste de cliente-servidor (Teste CS 0) identifica-se que o tempo de inicialização foi o segundo mais alto se forem considerados todos os testes. Porém, ainda dentro dos limites aceitáveis e de modo satisfatório. Ao analisar o resultado do Teste CS 0 apenas com os testes executados

dentro do Domínio *Tracker 1*, isto é, os Testes P2P 0 e 1 (os testes P2P 3 e 4 possuem o mesmo momento inicial do Teste P2P 1 e produzem resultados similares), constata-se que tanto o BitTorrent (Teste P2P 0) como a solução proposta (Teste P2P 1) apresentam desempenho melhor que o teste do cliente-servidor (Teste CS 0). Esse fato é consequência do comportamento padrão do VLC que logo após estabelecer a conexão (pode levar até 5 segundos) armazena (*buffer*) em média 5 segundos de conteúdo antes de iniciar a reprodução, já que a aplicação assume que haverá mais conteúdo a ser reproduzido e deseja evitar interrupções na reprodução por falta de conteúdo. A solução proposta também emprega uma versão modificada do VLC (seção 5.2.1) e apresenta um desempenho melhor essencialmente por usar blocos de 2 segundos, essa escolha faz com que o bloco seja reproduzido assim que é obtido. Contudo, caso o tamanho do bloco seja alterado para uma valor de 5 segundos um problema similar poderá ocorrer.

Quanto ao tempo médio para a Obtenção do Bloco (Tabela 9), os Testes P2P: 1, 3 e 4 apresentaram tempos similares e próximos do Teste P2P 0. Estes resultados são animadores porque o Teste P2P 0 emprega todos os *peers* disponíveis do sistema e obtém todos os blocos simultaneamente, enquanto os demais testes P2P empregam apenas dois *peers* e obtém os blocos sequencialmente e estão sujeitos a mudanças no decorrer da obtenção do conteúdo (caso dos Testes P2P: 3 e 4). Aqui, cabe destacar que, apesar do Teste P2P 1 não ter o menor resultado neste critério, o tempo obtido continua muito bom. Novamente o Teste P2P 2 apresentou o maior valor, porém isso é justificável dada a distância maior entre o *peer* solicitante e o conteúdo que está em um domínio diferente. O Teste CS 0 não possui blocos por obter o conteúdo através de *streaming* usando o protocolo RTP e desse modo essa métrica não pode ser aferida.

O Teste P2P 2 apresentou uma interrupção da apresentação do conteúdo em oito das dez execuções do experimento. Uma análise das interrupções revelou que nesse cenário o limite máximo do tempo para obtenção do bloco é de 1700ms, visto que após o Cliente obter o conteúdo através do P2P ainda são necessários 300ms para carregar

o conteúdo para a memória principal e o reproduutor apresentá-lo na tela. Embora uma breve interrupção não seja um problema grave é algo que afeta a experiência do usuário (QoE), algo crítico em sistemas de IPTV. A origem deste problema não está no sistema de localidade e sim na disposição dos conteúdos, sendo que caso os conteúdos sejam aprovados estratégicamente (e.g., via *Cache*) esse problema não irá ocorrer. De fato, os tempos obtidos nos demais Testes P2P (1, 3 e 4) estão bem abaixo do limite de 1700ms, permitindo concluir que o serviço de localidade obtém desempenho adequado nestes critérios.

Outro aspecto a ser considerado aqui é a capacidade de processamento do equipamento que o Cliente utiliza. Quanto mais rápido o Cliente for capaz de processar o conteúdo para reproduzi-lo, maior será o tempo limite para obter o conteúdo sem causar interrupção na sua apresentação. Sendo assim, o tempo de preparação de exibição no conteúdo também mostra-se importante para o correto dimensionamento de um sistema dessa natureza. No ambiente de testes utilizado, os computadores possuem uma capacidade de processamento alta e, dessa maneira, o tempo de preparação é mais reduzido do que se fossem empregados computadores de mesa tradicionais.

Deve-se notar que a avaliação sobre o desempenho da arquitetura de *trackers* hierárquicos apresentado na Tabela 9 tem, como critério básico, a garantia da execução correta do serviço de IPTV para os Clientes.

6.2.2 Controle do uso dos recursos de rede

É notável que a solução proposta consegue atingir um bom desempenho ao mesmo tempo que permite otimizar o uso dos recursos do sistema. No contexto deste trabalho, a otimização de recursos é obtida através da habilidade de controlar o fluxo de dados segundo a situação da rede e regras de negócio definidas pelo operador da rede. Buscando evidenciar a otimização obtida no Sistema IPTV, a Figura 43 mostra os resultados agrupados por enlace do Teste CS 0, Teste P2P 0 e Teste P2P 1.

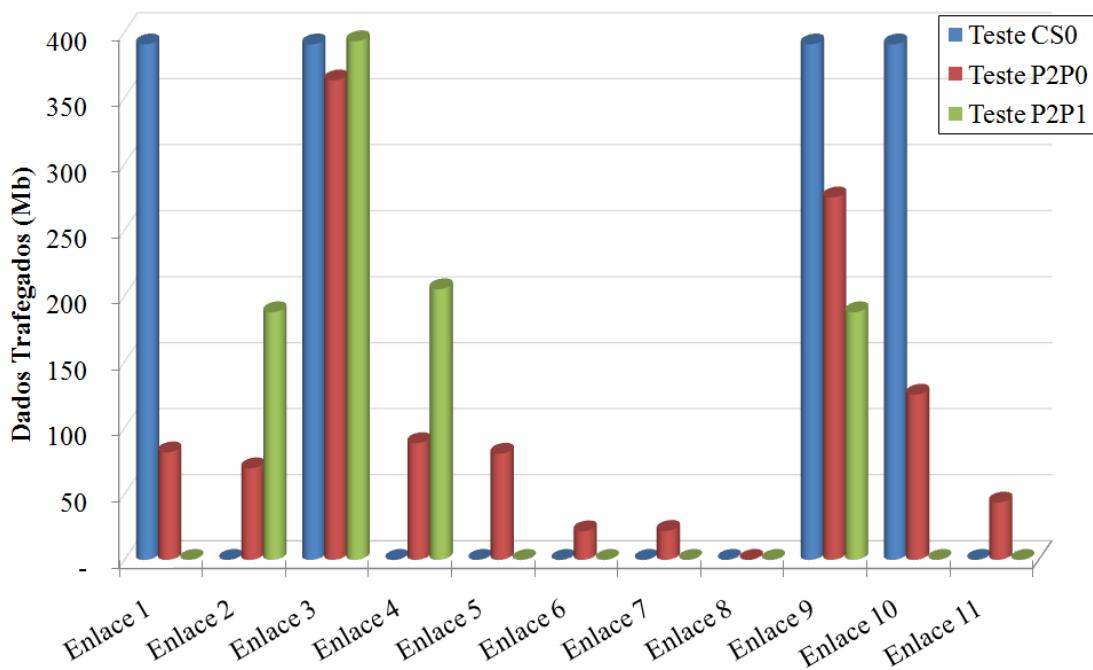


Figura 43: Comparação Teste CS 0, Teste P2P 0 e Teste P2P 1: Dados trafegados por enlace

A Figura 43 mostra que o BitTorrent padrão não considera os recursos da rede nem políticas definidas pelo seu operador para elaborar a lista *peers*, utilizando praticamente todos os enlaces do sistema (não usa o Enlace 8 por não haver dados disponíveis neste equipamento). O Teste P2P 1 demonstra que a mesma operação pode ser realizada usando menos enlaces e até com um desempenho melhor em vários dos critérios analisados. Entretanto, a Figura 43 não mostra a dinamicidade da solução e para esta finalidade os Testes P2P 3 e 4 são executados, demonstrando que mesmo aplicando modificações no sistema (quer seja pela condição da rede ou pelas políticas do operador da rede) o sistema pode continuar fornecendo os serviços de IPTV com desempenho satisfatório.

Os resultados do Teste CS 0 indicam o uso de um caminho (Enlace 3 - Enlace 9 - Enlace 10 -Enlace 1) apenas, o que pode levar a problemas de escalabilidade devido à limitação de capacidade nos enlaces do *backbone* (no cenário apresentado é representado pelo Enlace 10). Caso a quantidade de requisições aumentar consideravelmente, o Enlace 10 ficará sobrecarregado e a qualidade do serviço sofrerá degradação.

6.2.3 Análise de requisitos *versus* resultados

Uma vez que os testes foram executados e analisados, torna-se necessário avaliar se os requisitos foram satisfeitos. As Tabelas 10 e 11 descrevem cada requisito (seção 4.1.1) e informam se o mesmo é ou não satisfeito pela aplicação da solução P2PM no Sistema IPTV.

Tabela 10: Satisfação dos requisitos não funcionais

Requisito	Satisfeto	Observação
Permitir que um número considerável de usuários accesse os recursos com um desempenho satisfatório.	Parcialmente	O ambiente de testes utilizado possui uma quantidade limitada de <i>peers</i> , uma característica essencial das redes P2P é a escalabilidade e esta característica comprovada nas redes BitTorrent. A Solução P2PM possui várias características em comum com o BitTorrent, contudo, este requisito não pode ser aferido no protótipo em função da pequena quantidade de <i>peers</i> disponíveis no ambiente de testes. Contudo, umas das fraquezas das redes P2P é o seu desempenho com poucos <i>peers</i> e neste aspecto foi comprovado que a solução comporta-se bem: com apenas dois <i>peers</i> , ela consegue prover o conteúdo para a aplicação do Cliente satisfatoriamente.
Fornecer o conteúdo solicitado de modo a evitar, preferencialmente, a interrupção da reprodução do conteúdo após esta ser iniciada.	Sim	A análise dos resultados apresentada na Tabela 9 evidencia a satisfação desse requisito.
Fornecer recursos para monitorar as condições do sistema P2PM por parte do operador.	Sim	A interface de monitoração apresentada na seção 5.3 e no Apêndice C comprovam esse requisito. Além da Interface, o sistema de <i>logs</i> também auxilia no cumprimento desse requisito.

Tabela 11: Satisfação dos requisitos funcionais

Requisito	Satisfaito	Observação
Oferecer serviços de distribuição de conteúdo de vídeo empregando redes P2P de forma a otimizar o uso dos recursos da rede.	Sim	O uso do protocolo BitTorrent modificado através dos <i>trackers</i> hierárquicos demonstra a distribuição P2P e o grafo otimiza o uso dos recursos da rede. Os Testes P2P 1 e 2 demonstram que este requisito é satisfeito.
Possibilitar que o comportamento da distribuição P2P seja configurável e personalizável através de critérios definidos pelo operador do sistema.	Sim	O comportamento é configurável e personalizável através do grafo, conforme demonstrado no Teste P2P 4.
Identificar as condições da rede e usar essa informação para definir a forma como a distribuição P2P deve se comportar.	Sim	A atualização dinâmica do grafo satisfaz esse requisito, conforme demonstrado no Teste P2P 3
Permitir que sejam aplicadas regras de negócios na infraestrutura da rede de acordo com as suas condições.	Sim	A composição da fórmula que define o peso de cada aresta satisfaz esse requisito. No Sistema IPTV foi adotado o valor <i>Prioridade</i> para indicar a preferência dos enlaces pela regra de negócios e o Teste P2P 4 mostra que mesmo com as regras aplicadas a informação de rede foi utilizada através das alternâncias entre o <i>Peer3</i> e o <i>Proxy</i> na lista de <i>peers</i> .
Possibilitar a configuração de parâmetros do Sistema P2PM de modo a atender os requisitos de desempenho da aplicação-cliente que usa o conteúdo obtido a partir desse sistema.	Sim	Esse requisito é satisfeito pelo MGP que realiza a distribuição dos perfis de grafo na rede e permite definir as condições para a sua substituição.
Permitir o ajuste do comportamento da rede gerenciada através de perfis que definem como os recursos são utilizados e, consequentemente, impactam nos custos da rede.	Sim	O uso do critério <i>Prioridade</i> demonstra claramente como até mesmo variáveis simples podem ser aplicadas para modelar o comportamento da rede quanto a opções de custo.
Possibilitar a configuração dinâmica dos recursos controlados pela solução.	Sim	A inclusão de ou remoção de <i>peers</i> dinamicamente no grafo, sendo realizado pelo <i>trackers</i> responsáveis por cada domínio.
Minimizar o tráfego P2P nas redes <i>backbones</i> e redes regionais.	Sim	O método de busca por <i>peers</i> (Figura 11) prioriza <i>peers</i> dentro do domínio e evita os enlaces do <i>backbone</i> . A Figura 43 mostra que o Teste P2P 1 (usando o P2PM) prefere os enlaces locais no cenário empregado nos experimentos.

A análise de requisitos *versus* resultados comprova que os mesmos foram atingidos, com exceção do requisito não funcional de escalabilidade que necessita de uma simulação para a sua comprovação efetiva. Sendo assim, pode-se afirmar que os requisitos foram satisfeitos com base nos resultados aferidos no *testbed*.

6.3 Considerações do Capítulo

A primeira consideração a ser feita é que a arquitetura de *trackers* hierárquicos proposta, além de atender os requisitos definidos para ela, possui um desempenho melhor (dentre as métricas analisadas nessa tese) que um sistema BitTorrent padrão e um sistema cliente-servidor. Ademais, pode ser observado através dos testes tanto a flexibilidade como a granularidade para configurar os grafos e aplicá-los de acordo com a necessidade do operador da rede. Contudo, esse alto poder de personalização exige cuidados para não produzir comportamentos inesperados ou errôneos no sistema.

Os resultados obtidos no Teste P2P 2 permitiram concluir que, embora a proposta de *trackers* hierárquicos busque otimizar o uso dos recursos da rede, ainda é necessário avaliar questões como a topologia da rede, disposição do conteúdo e capacidade de processamento dos clientes para se obter resultados otimizados em termos de uso de recursos de rede aliados a melhoria ou manutenção do desempenho em comparação ao BitTorrent tradicional. Porém, vale ainda observar que estes aspectos necessitam ser avaliados de acordo com os requisitos da aplicação que irá usar a solução proposta para obter o conteúdo e os parâmetros de qualidade do serviço associado. Conclui-se desse modo, que não é um problema da solução proposta e sim uma questão de planejamento da aplicação da solução.

7 CONSIDERAÇÕES FINAIS

A solução proposta de “arquitetura de *trackers* hierárquicos para localidade em redes *peer-to-peer* gerenciadas” apresentada nesta tese atingiu os seus objetivos quanto ao fornecimento de serviços de conteúdo com necessidades específicas de desempenho que leva em consideração: aspectos de localidade, uso dos recursos da rede e políticas de uso desses recursos. A validação foi obtida a partir da aplicação da solução a um sistema de IPTV e pela análise dos resultados obtidos, que comprovaram a eficiência da solução e satisfação dos requisitos. Apenas um dos requisitos não funcionais foi parcialmente atendido, mais precisamente no que se refere à escalabilidade da solução. Esse requisito foi considerado parcialmente atendido, nesse momento, em decorrência que a escalabilidade não poder ser avaliada com propriedade devido ao número limitado de *peers* disponíveis no ambiente de teste. Contudo, o uso do BitTorrent como solução base possibilitou aproveitar-se da escalabilidade já identificada do BitTorrent na distribuição de conteúdo na Internet, principalmente em situações nas quais há um conjunto pequeno de conteúdo que é compartilhado por uma quantidade elevada de *peers*. As redes P2P tendem a apresentar o seu desempenho degradado quando possuem poucos *peers* e melhorado à medida que aumenta a quantidade de *peers*. Neste sentido, a solução proposta mostra-se positiva porque já apresenta um desempenho satisfatório com uma quantidade mínima de *peers*.

As modificações realizadas nos clientes BitTorrent (*seeders* e *leechers*) para incluir o serviço de localidade não afetaram o comportamento da obtenção do conteúdo mas sim a origem dos conteúdo (lista de *peers* retornada) e o acesso aos mesmos (definição

dos caminhos através dos grafos). Uma mudança que houve no cliente BitTorrent foi a inclusão dos dados dos conteúdos disponíveis nos *peers* em algumas mensagens de controle já existentes (e.g., *keep alive*) para o *tracker* e que ocorrem periodicamente, bem como a criação de uma nova mensagem para a notificação dos conteúdos (ou suas partes) disponíveis no *peer* logo após a sua obtenção. Outra mudança que pode ser citada é quanto a identificação dos *peers*, no BitTorrent essa identificação é dada por um número aleatório e no sistema proposto é uma ID que deve ser fornecida pelo provedor. Essa abordagem foi adotada diante da preocupação que os provedores de conteúdo têm em identificar quem possui o conteúdo ativo, aspecto que não é obtido através das identificações com números aleatórios do BitTorrent.

As mudanças mais profundas ocorreram no *tracker*, que na solução proposta assume um papel vital para o serviço de localidade. Além disso, os *trackers* da solução proposta possuem conhecimento das partes do conteúdo disponíveis em todo o sistema, informação que não está disponível no *tracker* BitTorrent tradicional. A informação de conteúdo nos *trackers* é fundamental para a elaboração de uma lista de *peers* adequada, fato que, aliado ao uso de grafos e níveis hierárquicos, torna a solução simples, robusta e propicia a escalabilidade. Outra mudança neste sentido é que na solução proposta existem dois tipos de *tracker* distintos: *trackers* de domínio e *Root Tracker*, enquanto no BitTorrent só há um tipo de *tracker*.

A criação da estrutura hierárquica de *trackers* implicou na necessidade da troca de mensagens entre *trackers*, algo que não existe no BitTorrent original. Neste sentido foram criadas mensagens usando *web services* de forma a possibilitar que outros módulos possam ser incluídos no futuro sem a necessidade de ser estudada a documentação completa das mensagens BitTorrent. Adicionalmente, o MGP e a coleta de dados nos elementos de rede (e.g., roteadores) também podem ser listados como características exclusivas da solução.

Um aspecto que merece destaque no protótipo empregado nos testes da solução

proposta (Sistema IPTV com serviço de localidade) é o uso de informações em tempo real (interface de monitoração) para analisar o comportamento do sistema. A interface de monitoração em tempo real proporcionou uma forma aprimorada de analisar os experimentos e mostrar a capacidade do sistema.

A elaboração da fórmula utilizada para computar os pesos nas arestas também merece ênfase. Embora, a solução proposta seja personalizável, o desenvolvimento das fórmulas para obter o comportamento esperado no protótipo mostrou-se desafiador e demandou um esforço considerável. As fórmulas somente foram definidas após uma avaliação da topologia da rede e uma definição clara da importância de cada valor empregado, pois os pesos das arestas foram projetados para serem sensíveis tanto à mudança do valor de *Prioridade* (Exemplo de regra de negócios) como condições da rede. Caso os valores definidos pelo operador (valor *Prioridade*, nesta tese) não sejam ponderados em relação às demais informações (nesta tese foi empregado a largura de banda disponível), eles podem não surtir o comportamento adequado. Como exemplo, pode-se mencionar uma soma simples dos critérios para definir o peso de uma aresta (peso da aresta = *Prioridade* + largura de banda disponível em bits), o que resulta no fato do valor *Prioridade* ter pouca importância na definição final do peso.

Durante a especificação da solução também foram consideradas algumas questões de disponibilidade da solução. Nesse sentido, foi identificado que o *tracker* pode ser interpretado como um ponto simples de falha (*Single Point of Failure* (SPF)). Porém, o fato dos *trackers* serem mantidos pelo operador da rede implica na possibilidade de soluções de contingência tradicionais (e.g., redundância e *cluster*). Este fato também ocorre no BitTorrent tradicional, cuja carga nos servidores é considerável porém não representa um problema crítico.

Outro fator a ser considerado na solução proposta refere-se ao fato dos *trackers* conhecerm os conteúdos e suas partes disponíveis nos *peers*, o que pode ser interpretado como um problema de privacidade apesar de também ser uma característica chave para

as organizações adotarem sistemas P2P para a distribuição do conteúdo. Mais precisamente, as organizações necessitam saber onde os conteúdos estão armazenados para que possam gerenciá-los, no sentido de removê-los quando isso for necessário ou até mesmo obter informações para otimizar o sistema. Sob a perspectiva dos clientes (*peers*) este fato pode ser considerado uma invasão de privacidade e poderia causar uma baixa adoção no sistema, apesar das operadoras de IPTV atuais também possuírem uma certa rastreabilidade do comportamento dos seus usuários. O tratamento desta e de outras questões de segurança não tratadas no sistema proposto pode, portanto, constituir-se como um tema interessante de trabalhos futuros na área.

7.1 Contribuições e Inovações

O desenvolvimento da tese, realizado dentro do contexto de dois projetos de pesquisa do LARC/PCS/EPUSP em parceria com a Ericsson Research envolvendo o tema IPTV, propiciou as seguintes contribuições e inovações, que em sua maioria já foram publicadas na forma de artigos científicos ou patentes (uma lista completa é apresentada no Apêndice A):

- Taxonomia de localidade para algoritmos P2P. Artigos:
 - Miers, C.; Simplício, M.; Gallo, D.; Carvalho, T; Bressan, G.; Souza, V.; Damola A. *Uma taxonomia para algoritmos de localidade em redes peer-to-peer*. Latin America Transactions, IEEE (Revista IEEE America Latina), doi:10.1109/TLA.2010.5595121., vol. 8, no. 4, pags. 323–331, 2010.
 - Miers, C.; Simplício, M.; Gallo, D.; Carvalho, T; Bressan, G.; Souza, V.; Damola A. *A taxonomy for locality algorithms on peer-to-peer networks*. Proceedings 8th International Information and Telecommunication Technologies Symposium (I2TS), pags. 61–67, Florianópolis/Brasil, 2009.

- Arquitetura de *trackers* hierárquicos para localidade em redes *peer-to-peer* gerenciadas. Patente submetida e artigo:
 - Miers, C.; Souza, V.; Carvalho, T. *System and method for managing data delivery in a peer-to-peer network*. European Patent Office, The Hague. 2010. PCT/EP2010/056996.
 - Miers, C.; Simplício, M.; Goya, W.; Carvalho, T; Souza, V. *An architecture for P2P locality in managed networks using hierarchical trackers*. Proceedings IEEE International Conference on Network and Service Management (CNSM), doi:10.1109/CNSM.2010.5691307, pags. 206–213, Niagara Falls/Canada, 2010.

Além da arquitetura outros aspectos seus são também inovadores:

- Uma estrutura para organizar *trackers* hierarquicamente;
- Um sistema para organizar os *peers* em domínios e subordiná-los a um *tracker* de domínio;
- Um modelo de tabela para agregar informações sobre conteúdos disponíveis tanto nos *peers* como nos domínios *tracker*;
- Um método de notificação de novos conteúdos, ou suas partes, para os *trackers*;
- Um método de configuração de um grafo de topologia no qual os elementos principais são pré-definidos e os *peers* são adicionados ou removidos dinamicamente;
- Um método para fornecimento de listas de *peers* otimizadas quanto ao caminho de rede;
- Um método para identificação dos conteúdos disponíveis nos *peers* pelo *tracker* responsável pelo domínio;

- Um método de agregação do conteúdo disponível nos *peers* de forma hierarquizada e que possibilita a busca estruturada dos *peers* que possuem determinado conteúdo;
- Um método para comunicação entre *trackers* para buscar, identificar e localizar tanto conteúdos como *peers*;
- Um método de um *tracker* repassar uma lista de *peers* na qual estão contidos *peers* que não estão disponíveis no *swarm* do domínio do solicitante;
- Aplicação de grafos configuráveis por aresta para modelar o comportamento dos *peers* de forma que influencie na rede física.

7.2 Trabalhos Futuros

A solução proposta possui diversas oportunidades de trabalho futuros, sendo elas:

- Realizar um estudo sobre a inclusão de mecanismos de segurança na solução proposta. No ano de 2012 foi submetido um projeto a FAPESP (Processo 2011/21592-8) para atender essa proposta, sendo que o mesma foi aprovada em Julho/2012 e iniciou em Agosto/2012.
- Implementar um mecanismo de controle de fluxo que interrompa a obtenção do conteúdo quando a área de armazenamento (*buffer*) do cliente atingir um quantidade de blocos pré-definidas. Na versão atual do sistema os blocos são obtidos sucessivamente um após o outro até todo o conteúdo ser adquirido. Na seção 6.1.1 foi identificado que softwares que empregam o modelo cliente-servidor (e.g., VLC) podem fazer um controle de fluxo para evitar obter partes do conteúdo que o cliente pode não assistir.
- Analisar a seleção de novos critérios e o seu uso na criação de fórmulas para computar o peso nas arestas.

- Implementar uma simulação para comprovar a escalabilidade do sistema, testar cenários e validar fórmulas. Um trabalho nesse sentido já foi iniciado usando como base o simulador de BitTorrent EBitSim (EVANGELISTA et al., 2011).
- Usar a solução proposta em conjunto com computação em nuvem para criar uma solução do tipo *Cloud Assisted P2P* (e.g., (JIN; KWOK, 2010)).
- Estudar a usos da solução proposta para outras aplicações (distribuição de software) ou cenários (computação móvel).
- Possibilitar a interoperabilidade controlada com sistemas BitTorrent e similares.

REFERÊNCIAS

- AGARWAL, S.; SINGH, J. P.; DUBE, S. Analysis and implementation of gossip-based p2p streaming with distributed incentive mechanisms for peer cooperation. *Adv. MultiMedia*, Hindawi Publishing Corp., New York, NY, United States, v. 2007, p. 8:1–8:12, April 2007. ISSN 1687-5680. Disponível em: <<http://dx.doi.org/10.1155/2007/84150>>.
- AHUJA, R. K. et al. Faster algorithms for the shortest path problem. *J. ACM*, v. 37, n. 2, p. 213–223, 1990.
- ANNAPUREDDY, S.; FREEDMAN, M. J.; MAZIERES, D. Shark: scaling file servers via cooperative caching. In: *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design and Implementation*. Berkeley, CA, USA: USENIX Association, 2005. p. 129–142.
- ANNAPUREDDY, S. et al. Is high-quality vod feasible using p2p swarming? In: *Proceedings of the 16th international conference on World Wide Web*. New York, NY, USA: ACM, 2007. (WWW '07), p. 903–912. ISBN 978-1-59593-654-7. Disponível em: <<http://doi.acm.org/10.1145/1242572.1242694>>.
- BALAKRISHNAN, V. K. *Schaum's Outline of Graph Theory: Including Hundreds of Solved Problems*. 1. ed. [S.l.]: McGraw-Hill, Inc., 1997. (Schaum's Outline). ISBN 978-0070054899.
- BARBOSA, M. W. et al. Using locality of reference to improve performance of peer-to-peer applications. In: *Proc. of the 4th international workshop on Software and performance*. California/USA: ACM, 2004. p. 216–227. ISBN 1-58113-673-0.
- BELLMAN, R. On a routing problem. *Quarterly of Applied Mathematics*, v. 16, p. 90, 87, 1958.
- BINZENHÖFER, A.; LEIBNITZ, K. Estimating churn in structured p2p networks. In: *International Teletraffic Congress*. [S.l.: s.n.], 2007. p. 630–641.
- BOUTABA, R. Peer-to-peer networking: State of the art and research challenges. In: *IEEE Network Operations and Management Symposium, 2008. NOMS 2008*. [S.l.: s.n.], 2008. p. xxxii.
- CAI, H.; WANG, J. Exploiting geographical and temporal locality to boost search efficiency in peer-to-peer systems. *IEEE Transactions on Parallel and Distributed Systems*, IEEE Computer Society, Los Alamitos, CA, USA, v. 17, n. 10, p. 1189–1203, 2006. ISSN 1045-9219.

- CAIDA. *Analyzing UDP usage in Internet traffic*. 01 2010. <Http://www.caida.org/research/traffic-analysis/tcpudpratio/>. Disponível em: <<http://www.caida.org/research/traffic-analysis/tcpudpratio/>>.
- CEBALLOS, M.-R.; GORRICO, J.-L. P2p file sharing analysis for a better performance. In: *Proceedings of the 28th international conference on Software engineering*. New York, NY, USA: ACM, 2006. (ICSE '06), p. 941–944. ISBN 1-59593-375-1. Disponível em: <<http://doi.acm.org/10.1145/1134285.1134458>>.
- CHEHAI, W.; XIANLIANG, L.; HANCONG, D. Analysis of content availability optimization in bittorrent. In: *Proceedings of the 2006 International Conference on Hybrid Information Technology - Volume 01*. Washington, DC, USA: IEEE Computer Society, 2006. (ICHIT '06), p. 525–529. ISBN 0-7695-2674-8. Disponível em: <<http://dx.doi.org/10.1109/ICHIT.2006.76>>.
- CHEN, Y. et al. When is P2P technology beneficial for IPTV services? In: *Proceedings of the 17th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'07)*. Illinois, USA: ACM, 2007.
- CHO, K. et al. The impact and implications of the growth in residential user-to-user traffic. In: *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2006. (SIGCOMM '06), p. 207–218. ISBN 1-59593-308-5. Disponível em: <<http://doi.acm.org/10.1145/1159913.1159938>>.
- CHOE, Y. R. et al. Improving vod server efficiency with bittorrent. In: *MULTIMEDIA '07: Proceedings of the 15th international conference on Multimedia*. New York, NY, USA: ACM, 2007. p. 117–126. ISBN 978-1-59593-702-5.
- COHEN, B. *Incentives Build Robustness in BitTorrent*. 2003. Disponível em: <<http://citeseer.ist.psu.edu/cohen03incentives.html>>.
- _____. *The BitTorrent Protocol Specification*. 2008. http://www.bittorrent.org/beps/bep_0003.html.
- DARLAGIANNIS, V.; MAUTHE, A.; STEINMETZ, R. Overlay design mechanisms for heterogeneous large-scale dynamic p2p systems. *Journal of Network Systems Management*, v. 12, n. 2, 2004. <http://dblp.uni-trier.de/db/journals/jnsm/jnsm12.html#DarlagiannisMS04>.
- DAS, S.; KANGASHARJU, J. Evaluation of network impact of content distribution mechanisms. In: *Proceedings of the 1st international conference on Scalable information systems*. Hong Kong: ACM, 2006. p. 35. ISBN 1-59593-428-6. Disponível em: <<http://portal.acm.org/citation.cfm?id=1146882>>.
- DIJKSTRA, E. A note on two problems in connexion with graphs. *Numerische Mathematik*, v. 1, n. 1, p. 271, 269, dec 1959.
- DOBRESCU, M. et al. Controlling parallelism in a multicore software router. In: *Proceedings of the Workshop on Programmable Routers for Extensible Services of*

- Tomorrow*. New York, NY, USA: ACM, 2010. (PRESTO '10), p. 2:1–2:6. ISBN 978-1-4503-0467-2. Disponível em: <<http://doi.acm.org/10.1145/1921151.1921154>>.
- DOYEN, G.; NATAF, E.; FESTOR, O. A hierarchical architecture for a distributed management of p2p networks and services. In: SCHÖNWÄLDER, J.; SERRAT, J. (Ed.). *Ambient Networks*. Springer Berlin / Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3775). p. 257–268. ISBN 978-3-540-29388-0. 10.1007/11568285 22. Disponível em: <http://dx.doi.org/10.1007/11568285_22>.
- ERMAN, J. et al. Identifying and discriminating between web and peer-to-peer traffic in the network core. In: *Proceedings of the 16th international conference on World Wide Web*. New York, NY, USA: ACM, 2007. (WWW '07), p. 883–892. ISBN 978-1-59593-654-7. Disponível em: <<http://doi.acm.org/10.1145/1242572.1242692>>.
- Ernst-Desmulier, J. et al. A Peer-to-Peer approach for cache sibling. In: *Proceedings of the First International Conference on Distributed Frameworks for Multimedia Applications*. [S.l.]: IEEE Computer Society, 2005. p. 323–330. ISBN 0-7695-2273-4.
- EVANGELISTA, P. et al. Ebtsim: An enhanced bittorrent simulation using omnet++ 4. In: *Proceedings of the 2011 IEEE 19th Annual International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*. Washington, DC, USA: IEEE Computer Society, 2011. (MASCOTS '11), p. 437–440. ISBN 978-0-7695-4430-4. Disponível em: <<http://dx.doi.org/10.1109/MASCOTS-2011.46>>.
- FLENNER, R. *Java P2P unleashed*. Sams, 2003. (Unleashed Series). ISBN 9780672323997. Disponível em: <<http://books.google.com.br/books?id=xIb3bPpF2E4C>>.
- FLOYD, R. W. Algorithm 97: Shortest path. *Commun. ACM*, v. 5, n. 6, p. 345, 1962.
- FREEDMAN, M. J.; FREUDENTHAL, E.; MAZIERES, D. Democratizing content publication with coral. In: *NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*. Berkeley, CA, USA: USENIX Association, 2004. p. 18.
- FREEDMAN, M. J. et al. Geographic locality of ip prefixes. In: *Proceedings of the Internet Measurement Conference 2005 on Internet Measurement Conference*. Berkeley, CA: USENIX Association, 2005. p. 13.
- GALIL, Z.; MARGALIT, O. All pairs shortest distances for graphs with small integer length edges. *Inf. Comput.*, v. 134, n. 2, p. 103–139, 1997.
- GALLO, D. *Arquitetura de IPTV com Suporte à Apresentação Deslocada no Tempo Baseada em Distribuição Peer-to-Peer*. Dissertação (Mestrado) — Escola Politécnica de São Paulo, 2009.
- GALLO, D. et al. Time-Shift based on P2P: exploiting network resources for a hybrid IPTV architecture. In: *I2TS 2008*. [S.l.: s.n.], 2008. ISBN 978-85-89264-09-9.

- _____. A multimedia delivery architecture for iptv with p2p-based time-shift support. In: *CCNC'09: Proceedings of the 6th IEEE Conference on Consumer Communications and Networking Conference*. Piscataway, NJ, USA: IEEE Press, 2009. p. 447–448. ISBN 978-1-4244-2308-8. Disponível em: <<http://dx.doi.org/10.1109/CCNC.2009.4784884>>.
- GHANSHANI, P.; BANSAL, T. Oasis: A hierarchical emst based p2p network. In: PAL, A. et al. (Ed.). *IWDC*. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3741), p. 201–212. ISBN 3-540-30959-4. http://dx.doi.org/10.1007/11603771_24.
- GREENGRASS, J.; EVANS, J.; BEGEN, A. Not all packets are equal, part i: Streaming video coding and sla requirements. *Internet Computing, IEEE*, v. 13, n. 1, p. 70 –75, jan.-feb. 2009. ISSN 1089-7801.
- GUO, L. et al. Measurements, analysis, and modeling of bittorrent-like systems. In: *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*. Berkeley, CA, USA: USENIX Association, 2005. (IMC '05), p. 4–4. Disponível em: <<http://dl.acm.org/citation.cfm?id=1251086%-.1251090>>.
- HALEPOVIC, E.; DETERS, R. The jxta performance model and evaluation. *Future Gener. Comput. Syst.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 21, p. 377–390, March 2005. ISSN 0167-739X. Disponível em: <<http://dl.acm.org/citation.cfm?id=1077755%-.1077760>>.
- HAMRA, A. A.; FELBER, P. A. Design choices for content distribution in p2p networks. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 35, n. 5, p. 29–40, 2005. ISSN 0146-4833.
- HANDURUKANDE, S. B. et al. Peer sharing behaviour in the edonkey network, and implications for the design of server-less file sharing systems. In: *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*. New York, NY, USA: ACM, 2006. (EuroSys '06), p. 359–371. ISBN 1-59593-322-0. Disponível em: <<http://doi.acm.org/10.1145/1217935.1217970>>.
- HANKS, S. et al. *RFC1702 - Generic Routing Encapsulation over IPv4 networks*. 1994.
- HEI, X. et al. A measurement study of a large-scale p2p iptv system. *IEEE Transactions on Multimedia*, v. 9, p. 1672–1687, 2007. ISSN 1520-9210.
- HOWISON, J. An introduction to the literature on online reputation systems for the market management of peer to peer services (mmapps) project. 2003.
- IPOQUE. *The Impact of P2P File Sharing, Voice over IP, Skype, Joost, Instant Messaging, One-Click Hosting and Media Streaming such as YouTube on the Internet*. October 2007. 25 p. Disponível em: <http://www.ipoque.com/userfiles/file/internet_study_2007.pdf>.

- ISOHUNT. *BitTorrent Sites Statistics*. 10 2012.
<http://isohunt.com/stats.php?mode=btSites>. Disponível em: <<http://isohunt.com/stats.php?mode=btSites>>.
- JIN, X.; KWOK, Y.-K. Cloud assisted P2P media streaming for bandwidth constrained mobile subscribers. In: *Proceedings of the 2010 IEEE 16th International Conference on Parallel and Distributed Systems*. Washington, DC, USA: IEEE Computer Society, 2010. (ICPADS '10), p. 800–805. ISBN 978-0-7695-4307-9. Disponível em: <<http://dx.doi.org/10.1109/ICPADS.2010%-.78>>.
- JOHNSON, D. B. Efficient algorithms for shortest paths in sparse networks. *J. ACM*, v. 24, n. 1, p. 1–13, 1977.
- KALOGERAKI, V.; GUNOPULOS, D.; ZEINALIPOUR-YAZTI, D. A local search mechanism for peer-to-peer networks. In: . McLean, Virginia, USA: ACM, 2002. p. 300–307. ISBN 1-58113-492-4.
- KARAKAYA, M.; KÖRPEOGLU İbrahim; ULUSOY Özgür. Counteracting free riding in peer-to-peer networks. *Comput. Netw.*, Elsevier North-Holland, Inc., New York, NY, USA, v. 52, n. 3, p. 675–694, 2008. ISSN 1389-1286.
- KOBAYASHI, H. et al. A self-organizing overlay network to exploit the locality of interests for effective resource discovery in p2p systems. In: . [S.l.]: IEEE Computer Society, 2005. p. 246–255. ISBN 0-7695-2262-9.
- LABOVITZ, C. *CDN and Over-The_Top Traffic Data*. 05 2012. Disponível em: <<http://conferences.infotoday.com/documents/150/2012CDNSummit-Deepfield.pdf>>.
- LATTRE, A. d. et al. *VideoLAN Streaming Howto*. set. 2011.
[Http://www.videolan.org/doc/streaming-howto/en/](http://www.videolan.org/doc/streaming-howto/en/). Disponível em: <<http://www.videolan.org/doc/streaming-howto/en/>>.
- LEGOUT, A.; URVOY-KELLER, G.; MICHIARDI, P. Rarest first and choke algorithms are enough. In: *IMC '06: Proc. of the 6th ACM SIGCOMM conference on Internet measurement*. USA: ACM, 2006. p. 203–216. ISBN 1-59593-561-4.
- LEONARD, D.; RAI, V.; LOGUINOV, D. On lifetime-based node failure and stochastic resilience of decentralized peer-to-peer networks. In: *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. New York, NY, USA: ACM, 2005. (SIGMETRICS '05), p. 26–37. ISBN 1-59593-022-1. Disponível em: <<http://doi.acm.org/10.1145/1064212.1064217>>.
- LI, J. Peer-to-peer multimedia applications. In: *Proceedings of the 14th annual ACM international conference on Multimedia*. New York, NY, USA: ACM, 2006. (MULTIMEDIA '06), p. 3–6. ISBN 1-59593-447-2. Disponível em: <<http://doi.acm.org/10.1145/1180639.1180641>>.

- LIANG, J.; NAHRSTEDT, K. Dagstream: locality aware and failure resilient peer-to-peer streaming. In: CHANDRA, S.; GRIWODZ, C. (Ed.). *Multimedia Computing and Networking 2006*. SPIE, 2006. v. 6071, p. 60710L. Disponível em: <<http://dx.doi.org/10.1117/12.643342>>.
- LIAO, X. et al. Anysee: Peer-to-peer live streaming. In: *Proc. of INFOCOM*. [s.n.], 2006. Disponível em: <<http://www.cs.ust.hk/~liu/AnySee.p>>.
- LIU, H. H. et al. Optimizing cost and performance for content multihoming. *SIGCOMM Comput. Commun. Rev.*, v. 42, n. 4, p. 371–382, ago. 2012. ISSN 0146-4833. Disponível em: <<http://doi.acm.org/10.1145/2377677.2377753>>.
- LOGUINOV, D.; CASAS, J.; WANG, X. Graph-theoretic analysis of structured peer-to-peer systems: routing distances and fault resilience. *IEEE/ACM Trans. Netw.*, IEEE Press, Piscataway, NJ, USA, v. 13, p. 1107–1120, October 2005. ISSN 1063-6692. Disponível em: <<http://dx.doi.org/10.1109/TNET.2005.857072>>.
- LOHMAR, T.; KRITZNER, J. *Evaluation of Content Distribution Networks (CDN) for On-demand Content in IPTV Environments*. 116 p. Tese (Doutorado) — Rheinisch-Westfälische Technische Hochschule Aachen, 2008.
- MANZATO, D.; FONSECA, N. da. A channel switching scheme for iptv systems. In: *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*. [S.l.: s.n.], 2010. p. 1 –6. ISSN 1930-529X.
- MANZATO, D. A.; FONSECA, N. L. da. A comparison of channel switching schemes for IPTV systems. In: *2011 IEEE International Conference on Communications (ICC)*. [S.l.]: IEEE, 2011. p. 1–6. ISBN 978-1-61284-232-5.
- MIERS, C. et al. A taxonomy for locality algorithms on peer-to-peer networks. In: *I2TS 2009*. [S.l.: s.n.], 2009. v. 1, p. 61–67. ISBN 978-85-89264-11-2.
- MISCHKE, J.; STILLER, B. Stiller: A methodology for the design of distributed search. In: *in P2P Middleware; IEEE Network, Volume 18 No. 1, January-February 2004*. [S.l.: s.n.], 2004. p. 30–37.
- MOLINA, M. et al. *Deliverable DJ1.2.3 - Network Metric Report*. [S.l.], feb 2006. 73 p. Disponível em: <www.geant2.net/.../pdf/GN2-05-265v4-Deliverable_DJ1-2-3_Network_Metric_Report.pdf>.
- NEGLIA, G. et al. Availability in bittorrent systems. In: *INFOCOM 2007. 26th IEEE International Conference on Computer Communications*. IEEE. [S.l.: s.n.], 2007. p. 2216 –2224. ISSN 0743-166X.
- NII. *The Unpredictable Certainty - White Papers - 2000 Steering Committee, Commission on Physical Sciences, Mathematics and Applications, National Research Council*. [S.l.]: The National Academies Press, 1997. ISBN 9780309060363.
- NORROS, I. et al. Flash crowd in a file sharing system based on random encounters. In: *interperf '06: Proceedings from the 2006 workshop on Interdisciplinary systems approach in performance evaluation and design of computer & communications systems*. New York, NY, USA: ACM, 2006. p. 4. ISBN 1-59593-503-7.

- ORAM, A. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly, 2001. Hardcover. ISBN 059600110X. Disponível em: <<http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/059600110X>>.
- P2PEDUCATION. *List of Peer-to-Peer Applications*. 06 2012. <http://p2peducation.pbworks.com/w/page/8897427/FrontPage>. Disponível em: <<http://p2peducation.pbworks.com/w/page/8897427/FrontPage>>.
- PATHAN, A. K.; BUYYA, R. *A taxonomy and survey of content delivery networks*. Australia, feb 2007. 44 p. Disponível em: <<http://www.gridbus.org/reports/CDN-Taxonomy.pdf>>.
- PAUL, R. *Verizon embraces P4P, a more efficient peer-to-peer tech*. 2008. Disponível em: <<http://arstechnica.com/news.ars/post/20080314-verizon-embraces-p4p-a-more-efficient-peer-to-peer-tech.html>>.
- PAXSON, V. Towards a framework for defining internet performance metrics. In: *Proc. INET '96*. [S.l.: s.n.], 1996.
- _____. Rfc 2330 - framework for ip performance metrics. 1998. Disponível em: <<http://www.faqs.org/rfcs/rfc2330.html>>.
- PETERSON, J.; GURBANI, V.; MAROCCO, E. *Alto WG, Application Layer Traffic Optimization Working group*. 2010. [Http://tools.ietf.org/wg/alto/](http://tools.ietf.org/wg/alto/).
- POESE, I. et al. Enabling content-aware traffic engineering. *SIGCOMM Comput. Commun. Rev.*, v. 42, n. 5, p. 21–28, 2012. ISSN 0146-4833. Disponível em: <<http://doi.acm.org/10.1145/2378956.2378960>>.
- POUWELSE, J. et al. *A Measurement Study of the BitTorrent Peer-to-Peer File-Sharing System*. [S.l.], 04 2004. 8 p. Disponível em: <http://www.pds.twi.tudelft.nl/~pouwelse/bittorrent_measurements.pdf>.
- QIU, D.; SRIKANT, R. Modeling and performance analysis of bittorrent-like peer-to-peer networks. *SIGCOMM Comput. Commun. Rev.*, v. 34, p. 367–378, out. 2004. ISSN 1581138628. Disponível em: <<http://portal.acm.org/citation.cfm?id=1030194.1015508>>.
- RATNASAMY, S. et al. A scalable content-addressable network. *SIGCOMM Comput. Commun. Rev.*, v. 31, n. 4, p. 161–172, ago. 2001. ISSN 0146-4833. Disponível em: <<http://doi.acm.org/10.1145/964723.383072>>.
- REXFORD, J. Network protocols designed for optimizability. In: *Information Sciences and Systems, 2006 40th Annual Conference on*. [s.n.], 2006. p. 351–354. Disponível em: <[10.1109/CISS.2006.286491](https://doi.org/10.1109/CISS.2006.286491)>.
- _____. *Refactoring Network Control and Management: A Case for the 4D Architecture*. [S.l.], 2008. Disponível em: <<http://www.cs.princeton.edu/~jrex/papers/4D-report.pdf>>.
- RIPEANU, M. Peer-to-peer architecture case study: Gnutella network. In: . [S.l.: s.n.], 2001. p. 99–100.

- RUBENSTEIN, D.; SAHU, S. Can unstructured p2p protocols survive flash crowds? *IEEE/ACM Transactions Network*, IEEE Press, Piscataway, NJ, USA, v. 13, n. 3, p. 501–512, 2005. ISSN 1063-6692.
- SANDVINE. *Sandvine : Global Broadband Trends 1H 2012*. [S.l.], 2012. Disponível em: <http://www.sandvine.com/downloads/documents/Phenomena_1H_2012-Sandvine_Global_Internet_Phenomena_Report_1H_2012.pdf>.
- SHAIKH-HUSIN, N.; HANI, M.; SENG, T. G. Implementation of recurrent neural network algorithm for shortest path calculation in network routing. In: *Parallel Architectures, Algorithms and Networks, 2002. I-SPAN '02. Proceedings. International Symposium on*. [S.l.: s.n.], 2002. p. 313–317.
- SHANAHAN, K. P.; FREEDMAN, M. J. Locality prediction for oblivious clients. In: CASTRO, M.; RENESSE, R. van (Ed.). *IPTPS*. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3640), p. 252–263. ISBN 3-540-29068-0.
- SRI PANIDKULCHAI, K. *A measurement-driven approach to designing peer-to-peer systems*. 135 p. Tese (Doutorado) — Carnegie Mellon University, 2005.
- SRI PANIDKULCHAI, K.; MAGGS, B. M.; ZHANG, H. Efficient content location using interest-based locality in peer-to-peer systems. In: *INFOCOM*. [S.l.: s.n.], 2003.
- STADING, T.; MANIATIS, P.; BAKER, M. Peer-to-peer caching schemes to address flash crowds. In: *Revised Papers from the First International Workshop on Peer-to-Peer Systems*. [S.l.]: Springer-Verlag, 2002. p. 203–213. ISBN 3-540-44179-4.
- STOICA, I. et al. Chord: A scalable peer-to-peer lookup service for internet applications. In: . San Diego, California, United States: ACM, 2001. p. 149–160. ISBN 1-58113-411-8.
- STUTZBACH, D.; REJAIE, R. Understanding churn in peer-to-peer networks. In: *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2006. (IMC '06), p. 189–202. ISBN 1-59593-561-4. Disponível em: <<http://doi.acm.org/10.1145/1177080.1177105>>.
- TU, X. et al. Design and deployment of locality-aware overlay multicast protocol for live streaming services. In: JIN, H.; REED, D. A.; JIANG, W. (Ed.). *NPC*. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3779), p. 105–112. ISBN 3-540-29810-X.
- _____. Nearcast: A locality-aware p2p live streaming approach for distance education. *ACM Trans. Internet Technol.*, ACM, New York, NY, USA, v. 8, n. 2, p. 1–23, 2008. ISSN 1533-5399.
- VASSILEVSKA, V. Nondecreasing paths in a weighted graph or: how to optimally read a train schedule. In: *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*. San Francisco, California: Society for Industrial and Applied Mathematics, 2008. p. 465–472.

- VEGA, G. d. l. *Insights in Telco CDN - Telefonica*. New York: [s.n.], 05 2012. Disponível em: <<http://conferences.infotoday.com/documents/150/CDN2012-Telefonica.pdf>>.
- VENUGOPAL, S.; BUYYA, R.; RAMAMOHANARAO, K. A taxonomy of data grids for distributed data sharing, management, and processing. *ACM Comput. Surv.*, v. 38, p. 3, 2006.
- VESTAL, J. *Accelerating Content Delivery In Asia*. 05 2012. Disponível em: <<http://conferences.infotoday.com/documents/150/2012CDNSummit-Pacnet.pdf>>.
- VU, L. et al. Measurement of a large-scale overlay for multimedia streaming. In: *Proceedings of the 16th international symposium on High performance distributed computing*. New York, NY, USA: ACM, 2007. (HPDC '07), p. 241–242. ISBN 978-1-59593-673-8. Disponível em: <<http://doi.acm.org/10.1145/1272366.1272410>>.
- WANG, H.; ZIMMERMANN, R.; KU, W.-S. Aspen: an adaptive spatial peer-to-peer network. In: *Proceedings of the 13th annual ACM international workshop on Geographic information systems*. Bremen, Germany: ACM, 2005. p. 230–239. ISBN 1-59593-146-5.
- WARNER, T. *Start Over -Time Warner Cable - New York City*. 09 2012. [Http://www.timewarnercable.com/nynj/learn/cable/startover.html](http://www.timewarnercable.com/nynj/learn/cable/startover.html). Disponível em: <<http://www.timewarnercable.com/nynj/learn%20/cable/startover.html>>.
- WEN, U.; WANG, W.; YANG, C. Traffic engineering and congestion control for open shortest path first networks. *Omega*, v. 35, n. 6, p. 671–682, mar 2006. ISSN 0305-0483. Disponível em: <<http://www.sciencedirect.com/science/article/B6VC4-4KMYG65-2/2/070e058bcbfd95374e89675e0ffa24a8>>.
- XIE, H. et al. *P4P: Explicit Communications for Cooperative Control Between P2P and Network Providers*. [S.I.]: Verizon, 2008. 7 p. http://www.dcia.info/documents/P4P_Overview.pdf.
- XU, Z.; HU, Y. Exploiting spatial locality to improve peer-to-peer system performance. In: *Proceedings of the The Third IEEE Workshop on Internet Applications*. [S.I.]: IEEE Computer Society, 2003. p. 121. ISBN 0-7695-1972-5.
- XUE, G.-T.; YOU, J.-Y.; JIA, Z.-Q. An interest group model for content location in peer-to-peer systems. In: . [S.I.]: IEEE Computer Society, 2004. p. 306–309. ISBN 0-7695-2206-8.
- YAN, F.; ZHAN, S. A peer-to-peer approach with semantic locality to service discovery. In: JIN, H. et al. (Ed.). *GCC*. [S.I.]: Springer, 2004. (Lecture Notes in Computer Science, v. 3251), p. 831–834. ISBN 3-540-23564-7.
- YU, Y.; LEE, S.; ZHANG, Z.-L. Leopard: A locality aware peer-to-peer system with no hot spot. In: BOUTABA, R. et al. (Ed.). *NETWORKING*. [S.I.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3462), p. 27–39. ISBN 3-540-25809-4.

ZHANG, K. S. an H. *Content Location in Peer-to-Peer Systems: Exploiting Locality*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005. ISBN 0387243569.

ZHAO, B. Y.; JOSEPH, A. D.; KUBIATOWICZ, J. Locality aware mechanisms for large-scale networks. In: *In Proceedings of the FuDiCo 02*. [S.l.: s.n.], 2002. p. 229–238.

ZHAO, B. Y.; KUBIATOWICZ, J. D.; JOSEPH, A. D. *Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and*. Berkeley, CA, USA, 2001.

ZHOU, M.; DAI, Y.; LI, X. A measurement study of the structured overlay network in p2p file-sharing systems. *Adv. MultiMedia*, Hindawi Publishing Corp., New York, NY, United States, v. 2007, n. 1, p. 10–10, 2007. ISSN 1687-5680.

APÊNDICE A - PUBLICAÇÕES

Publicações relacionadas diretamente a tese:

- Journal:

–Miers, C.; Simplício, M.; Gallo, D.; Carvalho, T; Bressan, G.; Souza, V.; Damola A. *Uma taxonomia para algoritmos de localidade em redes peer-to-peer.* Latin America Transactions, IEEE (Revista IEEE America Latina), doi:10.1109/TLA.2010.5595121., vol. 8, no. 4, pags. 323–331, 2010.

- Patente submetida(aguardando aprovação):

–Miers, C.; Souza, V.; Carvalho, T. *System and method for managing data delivery in a peer-to-peer network.* European Patent Office, The Hague. 2010. PCT/EP2010/056996.

- Conferências:

–Evangelista, P.; Amaral, M.; Miers, C.; Goya, W.; Simplicio, M.; Carvalho, T.; Souza, V. *EbitSim: An Enhanced BitTorrent Simulation Using OMNeT++ 4.* Proceedings IEEE 19th International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), doi:10.1109/MASCOTS.2011.46, pags.437-440. Cingapura/Cingapura, 2011,

- Miers, C.; Simplício, M.; Goya, W.; Carvalho, T; Souza, V. *An architecture for P2P locality in managed networks using hierarchical trackers.* Proceedings IEEE International Conference on Network and Service Management (CNSM), doi:10.1109/CNSM.2010.5691307, pags. 206–213, Niagara Falls/Canada, 2010.
- Amaral, M.; Miers, C.; Carvalho, T. *Proposta de melhoria do Electronic Program Guide EPG para redes híbridas de IPTV.* Proceedings Jornada Peruana de Computacion (JPC2010), Trujillo/Peru, 2010.
- Miers, C.; Simplício, M.; Gallo, D.; Carvalho, T; Bressan, G.; Souza, V.; Damola A. *A taxonomy for locality algorithms on peer-to-peer networks.* Proceedings 8th International Information and Telecommunication Technologies Symposium (I2TS), pags. 61–67, Florianópolis/Brasil, 2009.
- Gallo, D.; Miers, C; Coroama, V.; Carvalho, T.; Souza, V.; Karlsson, P. *A multimedia delivery architecture for IPTV with P2P-based time-shift support.* Proceedings 6th IEEE Conference on Consumer Communications and Networking Conference (CCNC), doi:10.1109/CCNC.2009.4784884, pags. 447–448. Las Vegas/EUA, 2009.
- Gallo, D.; Coroama, V.; Miers, C; Carvalho, T.; Souza, V.; Karlsson, P. *Time-Shift Based on P2P: Exploiting Network Resources for a Hybrid IPTV Architecture.* Proceedings 7th International Information and Telecommunication Technologies Symposium (I2TS), Foz do Iguaçu/Brasil, 2008.

Prêmio recebido:

- Premiado com o “*Bronze Paper Award*” no 8th International Information and Telecommunication Technologies Symposium (I2TS2009) pelo artigo: “*A taxonomy for locality algorithms on peer-to-peer networks*“

Publicações não relacionadas diretamente a tese:

•Journal:

—Gonzalez, N.; Miers, C.; Redígolo, F.; Carvalho, T.; Simplicio, M.; Näslund, M.; Pourzandi, M. *A quantitative analysis of current security concerns and solutions for cloud computing (extended version)*, Journal of Cloud Computing: Advances, Systems and Applications, vol. 1, no. 1, doi:10.1186/2192-113X-1-11, pags. 18, 2012.

•Capítulo de livro:

—Marcondes, C.; Salvador, M.; Rothenberg, C. E.; Carvalho, T.; Cardoso, K.; Abelém, A.; Miers, C. *Um Survey de Sistemas de Controle e Monitoramento para Testbed Experimentais em Redes*, Anais do XXX Simpósiio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), pags. 60, Minicursos, 2012.

—Carvalho, T.; Miers, C.; Dominicini, C.; Redígolo, F. *Key Issues on Future Internet. New Network Architectures*, vol. 297, doi:10.1007/978-3-642-13247-6_14, pags 221-236. Studies in Computational Intelligence. Springer Berlin/Heidelberg. 2010.

•Conferências:

—Barros, M.; Miers, C.; Simplício, M.; Carvalho, T.; Mangs, J-E.; Melander, B.; Souza, V. *Flow-Based Programming as a solution for Cloud Computing requirements*, Proceedings 1st International Conference on Operations Research and Enterprise Systems (ICORES), Vilamoura/Portugal, 2012.

- Miers, C.; Barros, M.; Simplício, M.; Gonzalez, N.; Evangelista, P.; Goya, W.; Carvalho, T.; Mangs, J-E.; Melander, B.; Souza, V. *Using Trade Wind to sail in the clouds*, Proceedings 23rd International Conference on Parallel and Distributed Computing and Systems (PDCS), pags. 8, Dallas/EUA, 2011.
- Gonzalez, N.; Miers, C.; Redígolo, F.; Carvalho, T.; Simplicio, M.; Näslund, M.; Pourzandi, M. *A Quantitative Analysis of Current Security Concerns and Solutions for Cloud Computing*, Proceedings 3rd IEEE International Conference on Cloud Computing Technology and Science, doi:10.1109/CloudCom.2011.39 ,pags. 231–238, Atenas/Grécia, 2011.
- Gonzalez, N.; Miers, C.; Redígolo, F.; Carvalho, T.; Simplicio, M.; Näslund, M.; Pourzandi, M. *A Taxonomy Model for Cloud Computing Services*, Proceedings 1st International Conference on Cloud Computing and Services Science (CLOSER), vol. 1, pags. 56–65, doi:10.5220/0003384800560065, Noordwijkerhout/Holanda, 2011.

APÊNDICE B - ARQUIVO XML DOS PERFIS DOS GRAFOS

Neste apêndice estão contidos os arquivos XML que possuem as configurações dos perfis de grafos usados no testes descritos no Capítulo 6. Na seção B.1 são listadas as configurações iniciais usando no domínio do *Tracker1* nos testes: **Teste P2P 1**, **Teste P2P 2**, **Teste P2P 3**, **Teste P2P 4** e **Teste P2P 5**. Já na seção B.2 são listadas as novas configurações do domínio *Tracker1* e que são aplicadas pelo Módulo Gerenciador de Perfis (MGP) após a ocorrência do gatilho de tempo, no **Teste P2P 5**. As seções B.3 e B.4 listam as configurações do domínio *Tracker2* e do *Root Tracker*. Nestes domínios não sofreram nenhuma alteração de perfil durante a execução dos testes que os envolveram.

B.1 *Tracker1*

```

<?xml version='1.0' encoding="iso-8859-1" ?>
<!DOCTYPE graphml SYSTEM "graphml.dtd">
<!-- ORIGINAL -->

<graphml>

  <key id="priority" for="edge">
    <default>80</default>
  </key>

  <key id="formulaRoutertoRouter" for="edge">
    <default>
      <par>
        <par>
          <cte>125000000</cte>
          <op type="sub"/>
        <par>
          <par>
            <var>c_1.3.6.1.2.1.2.2.1.5.x</var>
            <op type="div"/>
            <cte>8</cte>
          </par>
          <op type="sub"/>
        <par>
          <par>
            <var>t2_1.3.6.1.2.1.2.2.1.16.x</var>
            <op type="sub"/>
            <var>t1_1.3.6.1.2.1.2.2.1.16.x</var>
          </par>
          <op type="div"/>
          <var>interval</var>
        </par>
      </par>
    </default>
    <op type="add"/>
  </key>
</graphml>
```

```

<var>priority</var>
</default>
</key>

<key id="formulaPeertoRouter" for="edge">
<default>
<par>
<par>
<cte>125000000</cte>
<op type="sub"/>
<var>upAvailable</var>
</par>
<op type="div"/>
<cte>100000</cte>
</par>
<op type="add"/>
<var>priority</var>
</default>
</key>

<key id="formulaRoutertoPeer" for="edge">
<default>
<par>
<cte>0</cte>
</par>
</default>
</key>

<key id="formulaRoutertoTrackerReference" for="edge">
<default>
<par>
<cte>0</cte>
</par>
</default>
</key>

<key id="interval" for="graph" />

<graph id="tm11" edgedefault="directed">
<desc>Testbed CCE - TM1</desc>

```

```

<data key="interval">10</data>

<node id="tracker2" type="trackerReference" />

<node id="r1" type="router">
  <ip value="192.168.188.41" mask="29" oidlastnumber = "6" />
  <ip value="192.168.188.49" mask="29" oidlastnumber = "2" />
  <ip value="192.168.188.65" mask="29" oidlastnumber = "0" />
</node>

<node id="r2" type="router">
  <ip value="192.168.188.1" mask="32" oidlastnumber = "2" />
  <ip value="192.168.188.17" mask="32" oidlastnumber = "3" />
  <ip value="192.168.188.33" mask="29" oidlastnumber = "4" />
  <ip value="192.168.188.42" mask="29" oidlastnumber = "5" />
</node>

<node id="s1" type="router">
  <ip value="192.168.188.1" mask="32" oidlastnumber = "2" />
  <ip value="192.168.188.1" mask="28" oidlastnumber = "2" />
</node>

<node id="s2" type="router">
  <ip value="192.168.188.17" mask="32" oidlastnumber = "3" />
  <ip value="192.168.188.17" mask="28" oidlastnumber = "3" />
</node>

<node id="50aa50" type="proxy" />

<node id="ca1a34" type="cache" />

<edge id="e0" source ="r1" target="tracker2" sourceport="192.168.188.65">
  <data key="priority">0</data>
</edge>

<edge id="e1" source="r1" target="r2" sourceport="192.168.188.41" targetport="192.168.188.42">
  <data key="priority">70</data>
</edge>
```

```

<edge id="e2" source="r2" target="r1" sourceport="192.168.188.42" targetport="192.168.188.41">
  <data key="priority">70</data>
</edge>

<edge id="e3" source="50aa50" target="r1" targetport="192.168.188.49">
  <data key="priority">80</data>
</edge>

<edge id="e4" source="r1" target="50aa50" sourceport="192.168.188.49">
  <data key="priority">0</data>
</edge>

<edge id="e5" source="ca1a34" target="r2" targetport="192.168.188.33">
  <data key="priority">45</data>
</edge>

<edge id="e6" source="r2" target="ca1a34" sourceport="192.168.188.33">
  <data key="priority">0</data>
</edge>

<edge id="e7" source="r2" target="s1" sourceport="192.168.188.1" targetport="192.168.188.1">
  <data key="priority">80</data>
</edge>

<edge id="e8" source="s1" target="r2" sourceport="192.168.188.1" targetport="192.168.188.1">
  <data key="priority">80</data>
</edge>

<edge id="e9" source="r2" target="s2" sourceport="192.168.188.17" targetport="192.168.188.17">
  <data key="priority">80</data>
</edge>

<edge id="e10" source="s2" target="r2" sourceport="192.168.188.17" targetport="192.168.188.17">
  <data key="priority">80</data>
</edge>

</graph>
</graphml>

```

B.2 *Tracker1* com prioridades alteradas

```

<?xml version='1.0' encoding="iso-8859-1" ?>
<!DOCTYPE graphml SYSTEM "graphml.dtd">
<!-- ORIGINAL -->

<graphml>

<key id="priority" for="edge">
  <default>80</default>
</key>

<key id="formulaRoutertoRouter" for="edge">
  <default>
    <par>
      <par>
        <cte>125000000</cte>
        <op type="sub"/>
      <par>
        <par>
          <var>c_1.3.6.1.2.1.2.2.1.5.x</var>
          <op type="div"/>
          <cte>8</cte>
        </par>
        <op type="sub"/>
      <par>
        <par>
          <var>t2_1.3.6.1.2.1.2.2.1.16.x</var>
          <op type="sub"/>
          <var>t1_1.3.6.1.2.1.2.2.1.16.x</var>
        </par>
        <op type="div"/>
        <var>interval</var>
      </par>
    </par>
  </default>
  <op type="add"/>
</key>

```

```

<var>priority</var>
</default>
</key>

<key id="formulaPeertoRouter" for="edge">
<default>
<par>
<par>
<cte>125000000</cte>
<op type="sub"/>
<var>upAvailable</var>
</par>
<op type="div"/>
<cte>100000</cte>
</par>
<op type="add"/>
<var>priority</var>
</default>
</key>

<key id="formulaRoutertoPeer" for="edge">
<default>
<par>
<cte>0</cte>
</par>
</default>
</key>

<key id="formulaRoutertoTrackerReference" for="edge">
<default>
<par>
<cte>0</cte>
</par>
</default>
</key>

<key id="interval" for="graph" />

<graph id="tm11" edgedefault="directed">
<desc>Testbed CCE - TM1</desc>

```

```

<data key="interval">10</data>

<node id="tracker2" type="trackerReference" />

<node id="r1" type="router">
  <ip value="192.168.188.41" mask="29" oidlastnumber = "6" />
  <ip value="192.168.188.49" mask="29" oidlastnumber = "2" />
  <ip value="192.168.188.65" mask="29" oidlastnumber = "0" />
</node>

<node id="r2" type="router">
  <ip value="192.168.188.1" mask="32" oidlastnumber = "2" />
  <ip value="192.168.188.17" mask="32" oidlastnumber = "3" />
  <ip value="192.168.188.33" mask="29" oidlastnumber = "4" />
  <ip value="192.168.188.42" mask="29" oidlastnumber = "5" />
</node>

<node id="s1" type="router">
  <ip value="192.168.188.1" mask="32" oidlastnumber = "2" />
  <ip value="192.168.188.1" mask="28" oidlastnumber = "2" />
</node>

<node id="s2" type="router">
  <ip value="192.168.188.17" mask="32" oidlastnumber = "3" />
  <ip value="192.168.188.17" mask="28" oidlastnumber = "3" />
</node>

<node id="50aa50" type="proxy" />

<node id="ca1a34" type="cache" />

<edge id="e0" source ="r1" target="tracker2" sourceport="192.168.188.65">
  <data key="priority">0</data>
</edge>

<edge id="e1" source="r1" target="r2" sourceport="192.168.188.41" targetport="192.168.188.42">
  <data key="priority">0</data>
</edge>
```

```
<edge id="e2" source="r2" target="r1" sourceport="192.168.188.42" targetport="192.168.188.41">
  <data key="priority">20</data>
</edge>

<edge id="e3" source="50aa50" target="r1" targetport="192.168.188.49">
  <data key="priority">20</data>
</edge>

<edge id="e4" source="r1" target="50aa50" sourceport="192.168.188.49">
  <data key="priority">0</data>
</edge>

<edge id="e5" source="ca1a34" target="r2" targetport="192.168.188.33">
  <data key="priority">100</data>
</edge>

<edge id="e6" source="r2" target="ca1a34" sourceport="192.168.188.33">
  <data key="priority">0</data>
</edge>

<edge id="e7" source="r2" target="s1" sourceport="192.168.188.1" targetport="192.168.188.1">
  <data key="priority">0</data>
</edge>

<edge id="e8" source="s1" target="r2" sourceport="192.168.188.1" targetport="192.168.188.1">
  <data key="priority">20</data>
</edge>

<edge id="e9" source="r2" target="s2" sourceport="192.168.188.17" targetport="192.168.188.17">
  <data key="priority">0</data>
</edge>

<edge id="e10" source="s2" target="r2" sourceport="192.168.188.17" targetport="192.168.188.17">
  <data key="priority">0</data>
</edge>

</graph>
</graphml>
```

B.3 *Tracker2*

```

<?xml version='1.0' encoding="iso-8859-1" ?>
<!DOCTYPE graphml SYSTEM "graphml.dtd">
<graphml>

  <key id="priority" for="edge">
    <default>100</default>
  </key>

  <key id="formulaRoutertoRouter" for="edge">
    <default>
      <par>
        <cte>10000000000</cte>
        <op type="div" />
      <par>
        <var>priority</var>
        <op type="mul" />
      <par>
        <par>
          <var>c_1.3.6.1.2.1.2.2.1.5.x</var> <!-- ifSpeed in bits per second -->
          <op type="div"/>
          <cte>8</cte>
        </par>
        <op type="sub"/>
      <par>
        <par>
          <var>t2_1.3.6.1.2.1.2.2.1.16.x</var> <!-- ifOutOctets -->
          <op type="sub"/>
          <var>t1_1.3.6.1.2.1.2.2.1.16.x</var> <!-- ifOutOctets -->
        </par>
        <op type="div"/>
        <var>interval</var>
      </par>
    </par>
  </par>
</default>
</key>

  <key id="formulaPeertoRouter" for="edge">
    <default>

```

```

<par>
  <cte>10000000000</cte>
  <op type="div" />
  <par>
    <var>priority</var>
    <op type="mul" />
    <var>upAvailable</var>
  </par>
</par>
</default>
</key>

<key id="formulaRoutertoPeer" for="edge">
<default>
  <par>
    <cte>0</cte>
  </par>
</default>
</key>

<key id="formulaRoutertoTrackerReference" for="edge">
<default>
  <par>
    <cte>0</cte>
  </par>
</default>
</key>

<key id="interval" for="graph" />

<graph id="tm11" edgedefault="directed">
  <desc>Testbed CCE - TM2</desc>
  <data key="interval">30</data>
  <node id="tracker1" type="trackerReference" />
  <<<<< .mine
  <node id="r1" type="router">
    <ip value="192.168.188.41" mask="32" oidlastnumber = "6" />
    <ip value="172.20.5.0" mask="24" oidlastnumber = "6" />

```

```
=====
<node id="r1" type="router">
  <ip value="192.168.188.65" mask="32" oidlastnumber = "2" />
  <ip value="192.168.188.49" mask="29" oidlastnumber = "2" />
</node>

<node id="s1" type="router">
  <ip value="192.168.188.65" mask="32" oidlastnumber = "2" />
  <ip value="192.168.188.65" mask="28" oidlastnumber = "2" />
>>>>> .r428
</node>

<edge id="e0" source="r1" target="tracker1" sourceport="192.168.188.49">
  <data key="priority">0</data>
</edge>

<<<<< .mine
<edge id="e0" source="r1" target="tracker1" sourceport="192.168.188.41">
=====

<edge id="e1" source="s1" target="r1" sourceport="192.168.188.65" targetport="192.168.188.65">
>>>>> .r428
  <data key="priority">100</data>
</edge>

<edge id="e2" source="r1" target="s1" sourceport="192.168.188.65" targetport="192.168.188.65">
  <data key="priority">100</data>
</edge>
</graph>
</graphml>
```

B.4 Root Tracker

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE graphml SYSTEM "mtmGraphml.dtd">
<graphml>
  <key id="priority" for="edge">
    <default>50</default>
  </key>
  <key id="formulaRoutertoRouter" for="edge" reference="source">
    <default>
      <par>
        <cte>1</cte>
        <op type="div" />
      <par>
        <var>priority</var>
        <op type="mul" />
      <par>
        <par>
          <var>c_1.3.6.1.2.1.2.2.1.5.x</var>
          <!-- ifSpeed in bits per second -->
          <op type="div" />
          <cte>8</cte>
        </par>
        <op type="sub" />
      <par>
        <par>
          <var>t2_1.3.6.1.2.1.2.2.1.16.x</var>
          <!-- ifOutOctets -->
          <op type="sub" />
          <var>t1_1.3.6.1.2.1.2.2.1.16.x</var>
          <!-- ifOutOctets -->
        </par>
        <op type="div" />
        <var>interval</var>
      </par>
    </par>
  </default>
  </key>
  <key id="formulaRoutertoTracker" for="edge" reference="source">
    <default>

```

```

<par>
  <cte>0</cte>
</par>
</default>
</key>

<key id="formulaTrackertoRouter" for="edge" reference="target">
<default>
<par>
  <cte>1</cte>
  <op type="div" />
<par>
  <var>priority</var>
  <op type="mul" />
<par>
<par>
  <var>c_1.3.6.1.2.1.2.2.1.5.x</var>
  <!-- ifSpeed in bits per second -->
  <op type="div" />
  <cte>8</cte>
</par>
  <op type="sub" />
<par>
<par>
  <var>t2_1.3.6.1.2.1.2.2.1.10.x</var>
  <!-- ifInOctets -->
  <op type="sub" />
  <var>t1_1.3.6.1.2.1.2.2.1.10.x</var>
  <!-- ifInOctets -->
</par>
  <op type="div" />
  <var>interval</var>
</par>
</par>
</par>
</par>
</default>
</key>

<key id="interval" for="graph" />
<graph id="tm11" edgedefault="directed">
  <desc>This graph is an example of a MTM Scope</desc>

```

```

<data key="interval">30</data>
<node id="r1" type="router">
  <ip value="172.20.5.1" mask="24" oidlastnumber="1" />
  <ip value="172.20.7.1" mask="24" oidlastnumber="3" />
</node>
<node id="r2" type="router">
  <ip value="172.20.5.2" mask="24" oidlastnumber="1" />
  <ip value="172.20.8.1" mask="24" oidlastnumber="2" />
</node>

<node id="t1" type="tracker" ipandport="172.20.5.11:8080"/>

<node id="t2" type="tracker" ipandport="172.20.5.2:8080"/>

<edge id="e1" source="r1" target="r2" sourceport="172.20.5.1" targetport="172.20.5.2">
  <data key="priority">40</data>
</edge>

<edge id="e2" source="r2" target="r1" sourceport="172.20.5.2" targetport="172.20.5.1">
  <data key="priority">40</data>
</edge>

<edge id="e3" source="r1" target="t1" sourceport="172.20.7.1">
  <data key="priority">0</data>
</edge>

<edge id="e4" source="t1" target="r1" targetport="172.20.7.1">
  <data key="priority">40</data>
</edge>

<edge id="e5" source="r2" target="t2" sourceport="172.20.8.1">
  <data key="priority">0</data>
</edge>

<edge id="e6" source="t2" target="r2" targetport="172.20.8.1">
  <data key="priority">40</data>
</edge>
</graph>
</graphml>

```

APÊNDICE C - INTERFACE DE MONITORAÇÃO EM TEMPO REAL

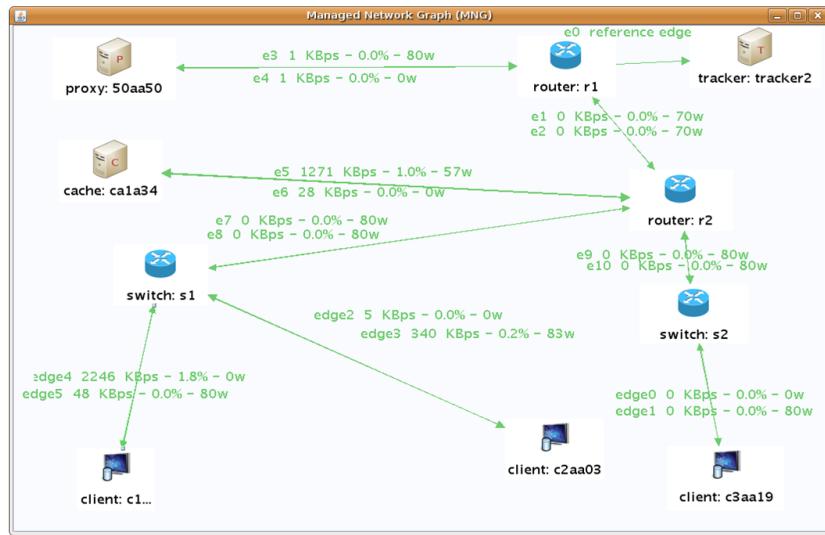


Figura 44: Interface de monitoração do sistema em tempo real

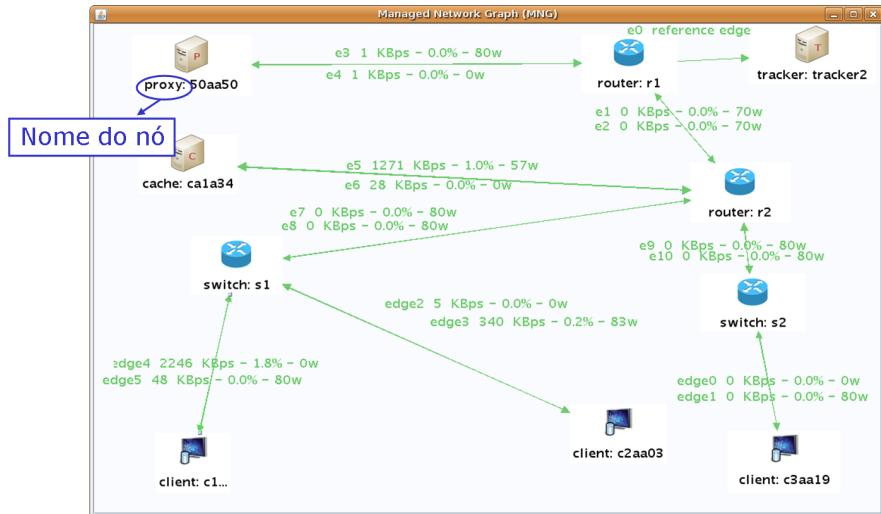


Figura 45: Interface de monitoração: Nome do Nô

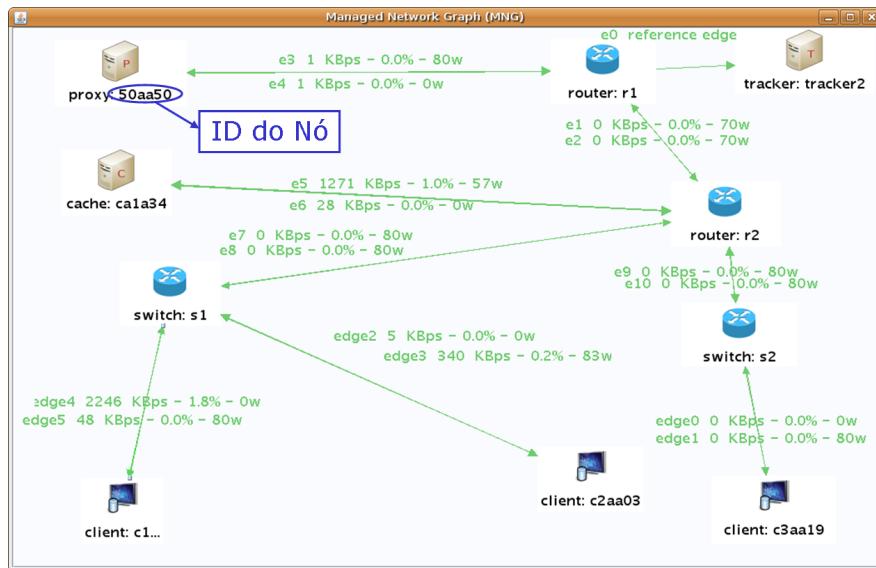


Figura 46: Interface de monitoração: ID do Nô

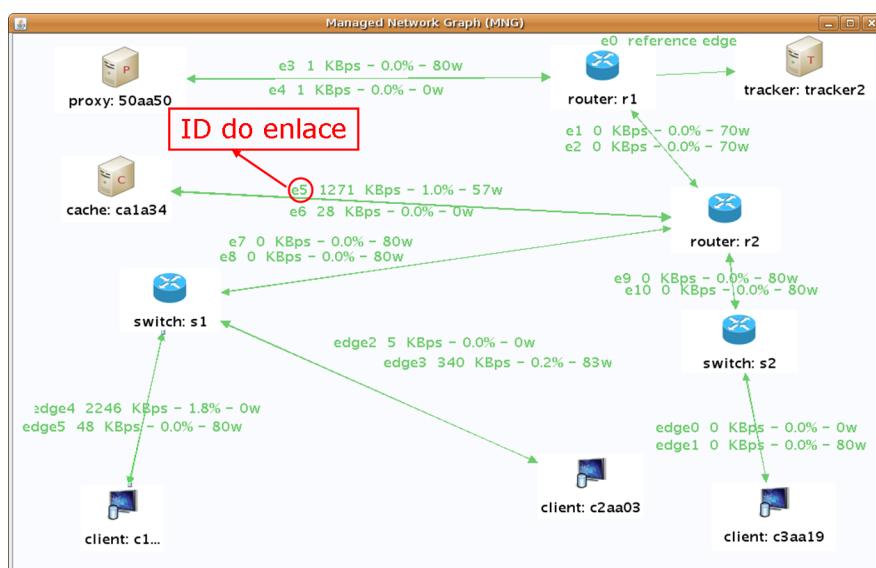


Figura 47: Interface de monitoração: Nome do enlace

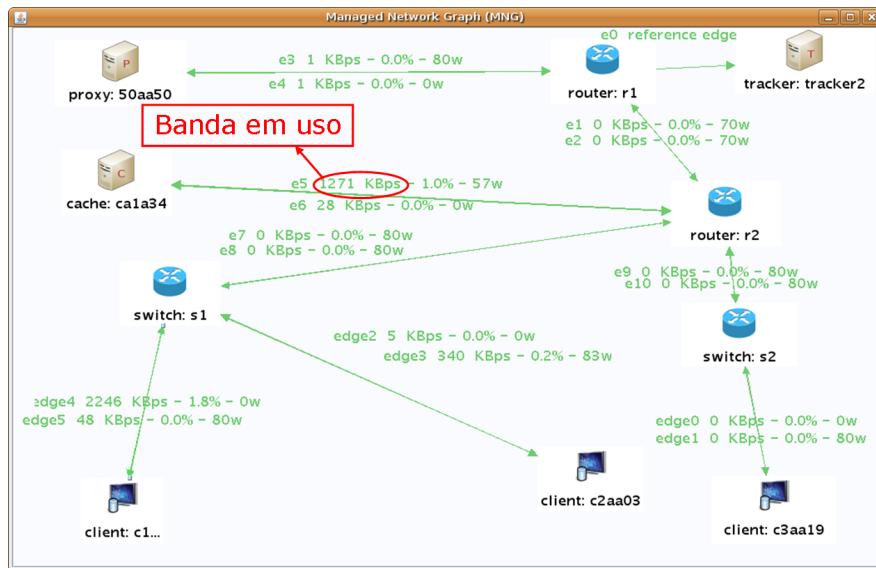


Figura 48: Interface de monitoração: Banda em uso

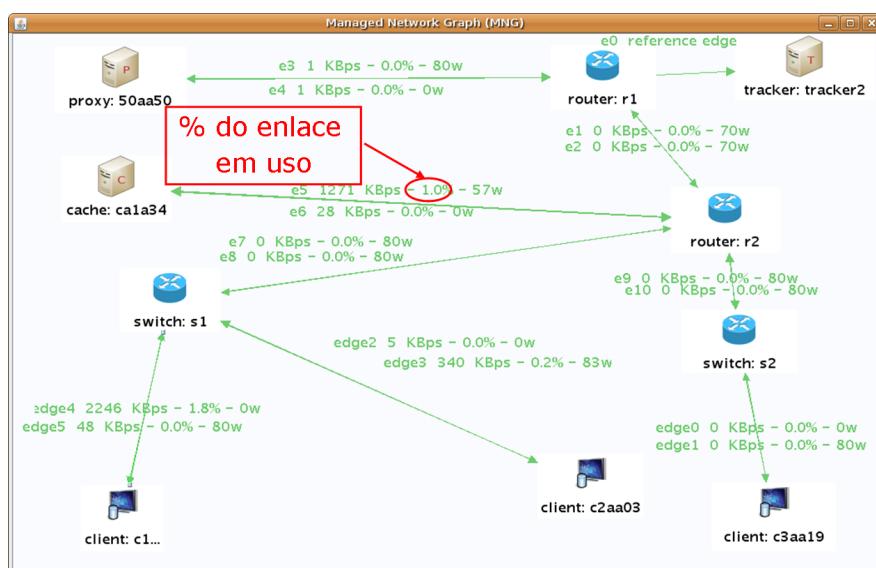


Figura 49: Interface de monitoração: % do enlace em uso

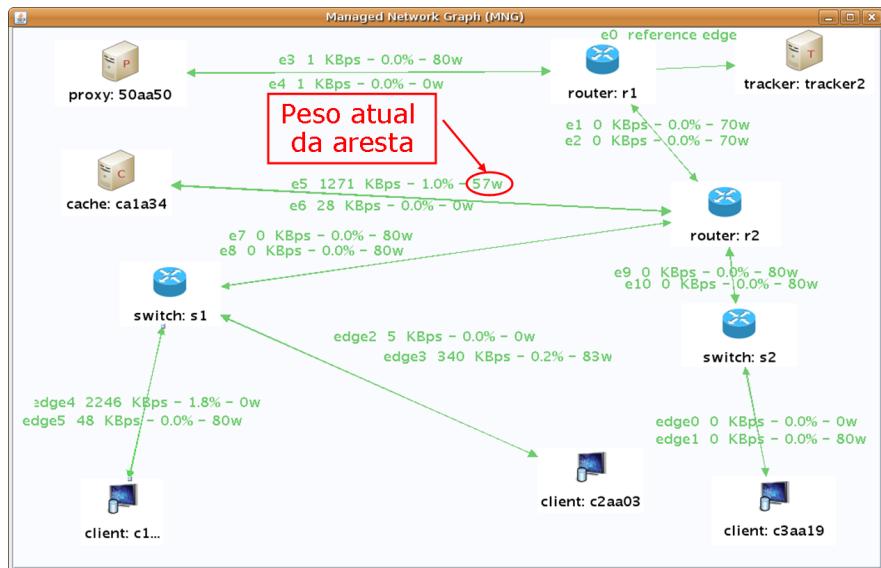
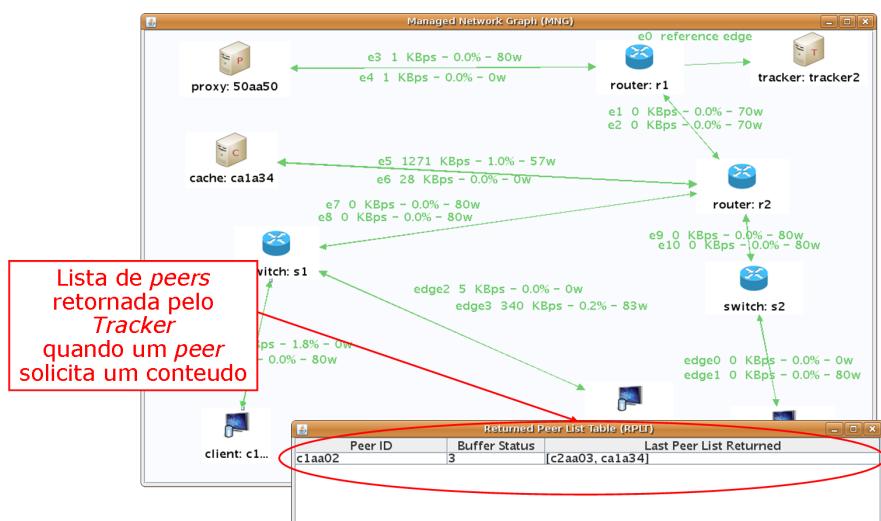


Figura 50: Interface de monitoração: peso atual da aresta

Figura 51: Interface de monitoração: listas de *peers* retornados

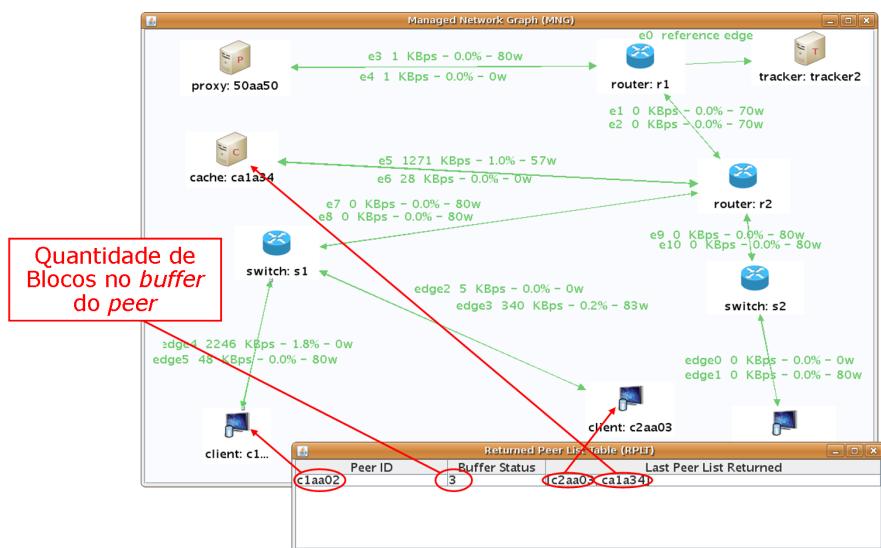


Figura 52: Interface de monitoração: quantidade de blocos no buffer do peer