Avaliação de Algoritmos para Seleção de Pares em Streaming BitTorrent Usando Regressão Linear e K-Means

Peron Rezende de Sousa[1], **Antonio Augusto de Aragão Rocha**[1], **José Viterbo Filho**[1]

[1]Instituto de Computação – Universidade Federal Fluminense (UFF)
Av. Gal. Milton Tavares de Souza, s/n – Boa Viagem
91.501-970 – Niterói – RJ – Brazil

{prezende, arocha, viterbo}@ic.uff.br

Abstract. *In this paper we compare 4 algorithms for peer selection and the traditional method of BitTorrent. For this purpose, we use the Linear Regression and K-Means. Research shows that the P4P and the S4Q are best suited for video transmission, since they have a good predictability with smoothing value of k=0.03. They also feature 3 to 4 quality levels well defined during the video stream.*

Resumo. *Neste trabalho comparamos 4 algoritmos para seleção de pares e o método tradicional do BitTorrent. Para esse fim, utilizamos a Regressão Linear e o K-Means. A pesquisa mostra que o P4P e o S4Q são os mais indicados para transmissão de vídeo, pois apresentam uma boa previsibilidade com smoothing value de k=0,03. Eles também apresentaram de 3 à 4 estágios de qualidade bem definidos durante a transmissão do vídeo.*

1. Introdução

A distribuição de vídeo (ou *streaming* de vídeo) tem crescido dia após dia e é uma das modalidades que mais consome os recursos da Internet. Além dos sistemas Par-a-Par (*peer-to-peer* - P2P), existem outras soluções para fornecimento de vídeo, como o modelo cliente-servidor, o *multicast* e as redes de distribuição de conteúdo (*Content Delivery Network - CDN*). Porém, nenhum desses modelos se compara ao P2P em custo e escalabilidade.

Assim como as demais soluções, o P2P nem sempre consegue oferecer determinados níveis de qualidade de serviço (*Quality of Service - QoS*). A transmissão de vídeo requer uma grande largura de banda passante e é sensível ao atraso, *jitter* e perda de pacotes. Garantir escalabilidade e valores adequados aos indicadores de QoS são grandes desafios para qualquer arquitetura que venha ser empregada [Moraes et al. 2008]. Quando se fala de sistemas P2P para transmissão de vídeo, um mecanismo exerce um papel fundamental na eficiência do seu funcionamento: a seleção de pares. Selecionar estrategicamente os melhores vizinhos pode ser essencial para se alcançar o nível de QoS requerido pela aplicação.

Diferentes mecanismos já foram propostos na literatura para tornar mais eficiente a seleção de pares. Para este trabalho escolhemos [Xie et al. 2008, Choffnes and Bustamante 2008, Polaczyk and Cholda 2010, de Sousa et al. 2014], por

terem sido desenvolvidos como *plugins* para uma mesma aplicação BitTorrent *open source* e multiplataforma, o Vuze [Vuze 2013]. Isso torna a comparação dos 4 algoritmos e do método tradicional, por meio da Regressão Linear e K-Means, totalmente imparcial.

Nosso objetivo avaliar o estado da arte em algoritmos para seleção de pares considerando a experiência do usuário na visualização de um vídeo, durante todo o processo de obtenção do conteúdo. Para esse fim escolhemos a Regressão Linear, pois ela pode mostrar o quão "previsível" é a QoE de cada algoritmo. Já o K-Means pode destacar agrupamentos que representam "estágios" de qualidade, ou seja, momentos em que a visualização pode ser melhor ou pior.

Este trabalho possui a seguinte organização: a Seção 2 discute os termos e as teorias que formam a base dessa pesquisa; a Seção 3 explica a definição da análise proposta para o uso da Regressão Linear e do K-Means e as informações que esperamos obter; a Seção 4 descreve o ambiente em que foram realizados os experimentos, apresenta os resultados e uma análise dos mesmos; por último, a Seção 5 descreve as conclusões e aponta possíveis trabalhos futuros.

2. Fundamentação Teórica

O BitTorrent é um protocolo P2P, criado por Bram Cohen, que tem o objetivo de favorecer o compartilhamento de conteúdo entre os usuários (*peers*), que fazem parte de um grupo (*swarm*). Esses usuários são qualificados em fornecedor do conteúdo (*seeder*), possuidor da lista de participates (*tracker*) e interessados no material (*leecher*) [Moraes et al. 2008].

Na distribuição, o arquivo (ou *bundle* de arquivos) que será compartilhado é dividido em pedaços (*chunks*), de normalmente 512 KBytes. Para cada um deles é calculado um código *hash* (SHA1) de 20 bytes correspondente ao seu conteúdo. Esse código serve para verificar a integridade do pedaço, após seu recebimento. A sequência de códigos *hash* e o endereço do *tracker* são informações obtidas no arquivo de metadados (.torrent). Com esses dados um *leecher* pode entrar em um *swarm*.

Ao entrar em um *swarm*, o *leecher* solicita um pedaço escolhido aleatóriamente (modo *Random First Piece*). Depois de receber esse pedaço ele passa a solicitar o mais raro (modo *Rarest First*). Por último, quando todos os pedaços já foram solicitados, o *leecher* dispara requisições do que falta a todos os outros *peers* (modo *Endgame*).

Repare que, seguindo a estratégia acima, os pedaços tendem a chegar de forma desordenada, prejudicando o interesse de quem tem a intenção de assistir um vídeo, com BitTorrent, durante o *download*. No decorrer deste trabalho vamos nos referir a essa estratégia como Algoritmo de Seleção Aleatória de Pedaços (ASAP).

Não demorou muito para surgir propostas à requisição ordenada de pedaços. Essa idéia consegue melhorar a reprodução de um vídeo durante sua transmissão, mas provoca uma maior latência e outros tipos de problemas. Neste trabalho vamos nos referir a essa estratégia como Algoritmo de Seleção Sequencial de Pedaços (ASSP).

Outro ponto fundamental no BitTorrent é a seleção de pares. O algoritmo tradicional não utiliza uma central de alocação de recursos, cada par é responsável pela maximização do seu *download* e escolha dos pares para os quais realizará *upload* seguindo a estratégia *tit-for-tat*. Nessa estratégia os pares não colaborativos são "afogados" (*choke*), ou seja, deixam de receber *upload*. De tempos em tempos um par afogado é escolhido

aleatoriamente para voltar a receber dados. Em resumo, os pares fornecem dados (*upload*) para bons fornecedores (*download*) e negam dados para quem não colabora (*free-riders*).

2.1. Algoritmos para Seleção de Pares

Nesta seção apresentamos 4 algoritmos de seleção de pares propostos na literatura são eles: **P4P**[Xie et al. 2008], **Ono**[Choffnes and Bustamante 2008], **Yukka**[Polaczyk and Cholda 2010] e **S4Q**[de Sousa et al. 2014].

Entre as soluções com os melhores resultados e mais referenciadas, está o **P4P**, que usa dados fornecidos pelos ISPs (*Internet Service Providers*) sobre suas redes para identificar pares melhores. Nem todos os ISPs têm suporte para P4P, portanto seu uso nem sempre representa vantagens atualmente, mas quando existe esse suporte ocorre uma melhora no desempenho para o usuário final e também há uma redução no tráfego entre ISPs, o que representa um custo menor para os provedores [Xie et al. 2008].

Também encontramos o **Ono** entre as mais difundidas. Sua estratégia é fazer uso das informações de redirecionamento provenientes das redes de distribuição de conteúdo (*Content Delivery Network - CDN*). Ele utiliza esses dados para identificar colegas próximos e potencialmente acelerar *downloads*, além de reduzir o trafégo entre ISPs. Segundo [Choffnes and Bustamante 2008], esse método consegue aumentar a taxa média de *download* em 31%.

Outra proposta é o método **Yukka**. Essa solução realiza consultas aos *Regional Internet Registries* (RIRs), para promover agrupamentos geográficos a partir da atribuição de uma nota de similaridade entre os pares. Os resultados em [Polaczyk and Cholda 2010] mostram uma redução de até 25% no tempo de *download*.

A proposta mais recente, intitulada Seleção por Qualidade (**S4Q**), avalia a qualidade percebida pelos dos participantes (*Quality of Experience* - QoE) de uma rede BitTorrent com base em uma nova metodologia, denominada Avaliação Áurea de Qualidade (A^2Q)[de Sousa 2013, de Sousa et al. 2014]. Essa métrica quantifica e qualifica o QoE através da ausência de pedaços do vídeo no momento da reprodução e o resultado é chamado de "nível de estresse". O S4Q, além de avaliar o QoE, também verifica, para cada participante, com quais outros participantes foi mantida uma comunicação continua ao longo do tempo, formando com isso uma Lista de Pares Estáveis (LPE). Com isso o QoE obtido é atribuído a LPE. Ao promover a troca de LPEs com boa QoE é possível realizar transmissões mais rápidas e com mínimo de interrupções. Essa solução não depende de fontes externas, mas depende do seu uso pelos outros pares.

2.2. Regressão Linear e K-Means

A Regressão Linear é um método para se estimar a condicional (valor esperado) de uma variável **y**, dados os valores de algumas outras variáveis **x**. A regressão, em geral, trata da questão de se estimar um valor condicional esperado. A Regressão Linear é chamada "linear" porque se considera que a relação da resposta às variáveis é uma função linear de alguns parâmetros. Com ela é possível desenhar uma linha reta que melhor se ajuste a posição dos pontos (*best-fit line*), mesmo quando os dados não formam uma disposição linear. Quando os pontos seguem uma determinada formação cuja previsibilidade não é bem representada por uma *best-fit line*, pois muitos pontos se encontram afastados da linha, temos o que é conhecido como erro médio quadrático (*mean-squared error*) e para

reduzir esse erro podemos usar uma técnica conhecida como *Locally Neighted Linear Regression* (LNLR). Nessa técnica um peso **k** (*smoothing value*) é utilizado para suavizar o erro. Quanto menor o valor, menor o erro e maior o esforço computacional. Um **k** igual a 1 tem o mesmo resultado da *best-fit line*. Neste trabalho, testamos vários valores para **k**, reduzindo gradativamente seu valor, até que um dos algoritmos apresentasse uma baixa margem de erro.

Agrupamento K-Means é um método de *clustering* que objetiva particionar **n** observações dentre **k** *clusters* onde cada observação pertence ao *cluster* mais próximo da média. O problema é computacionalmente difícil (NP-difícil), no entanto, existem algoritmos heurísticos eficientes que são comumente empregados e convergem rapidamente para um local *optimum*. Estes são geralmente semelhantes ao algoritmo de maximização da expectativa para misturas de distribuições gaussianas através de uma abordagem de refinamento iterativo utilizado por ambos os algoritmos. Além disso, ambos usam os centros de *clusters* para modelar dados, no entanto, a clusterização K-Means tende a encontrar *clusters* de extensão espacial comparáveis enquanto o mecanismo de maximização da expectativa permite ter diferentes formas [Harrington 2012].

3. Definição da Análise Proposta

Para avaliarmos a qualidade de um vídeo de uma maneira mais simples, decidimos nos apoiar na ausência de pedaços ao invés da ausência de quadros, como foi sugerido por [Rodríguez-Bocca 2008]. Para auxiliar nos cálculos precisamos chegar ao tempo de cada pedaço e, consequentemente, definir uma taxa de transmissão. O NetFlix, por exemplo, utiliza taxas de 512 Kbits/s (para baixa resolução) e 1.536 Kbits/s (para alta definição) [NetFlix 2013]. Por uma questão prática, neste trabalho optamos pela primeira. Considerando que o protocolo BitTorrent utiliza pedaços de 512 KBytes, podemos concluir que, em vídeos de baixa resolução, cada pedaço conterá cerca de 8 segundos. Como um vídeo, por padrão, trabalha a uma taxa de 23 quadros por segundo, cada um destes pedaços conterá cerca de $8 \times 23 = 184$ quadros.

A literatura é escassa quando se fala sobre dois aspectos em particular: (1) relações entre perdas de pedaços versus QoE; e, (2) estudos que pontuem a percepção desse evento fazendo uso de avaliações subjetivas baseadas em *Mean Opinion Score (MOS)*. [Krishnan and Sitaraman 2012] estudaram o impacto da qualidade na transmissão de vídeo utilizando extensivos *traces* da CDN Akamai, que incluem 23 milhões de visualizações feitas por 6,7 milhões de visitantes. Os autores mostraram que os espectadores abandonam o vídeo se este levar mais do que dois segundos para iniciar, para cada segundo adicional há um aumento de 8% na taxa de abandono. A pesquisa também mostra que uma quantidade moderada de interrupções pode diminuir significativamente o número de espectadores. Essas observações servem de inspiração foram cosideradas neste trabalho.

Para estabelecer uma relação entre o histórico de perda de pedaços e a QoE, definimos pontuações distintas para reproduções bem sucedidas e ausências de pedaços.

Contrariando o que foi observado na rede Akamai, decidimos iniciar a verificação das ausências 40 segundos após a solicitação de um vídeo. Esse tempo foi escolhido por ser o mesmo necessário à chegada dos primeiros dados e formação de *buffer* no AnySee, também por ser equivalente a 5 pedaços de vídeo, quando o mesmo possui uma taxa total

de 512 Kbits/s e está dividido em pedaços de 512 KBytes. Escolhemos o AnySee por ele ser um *player* de vídeo ao vivo e exigir apenas 40 segundos, enquanto que outros, como o CoolStreaming, necessitam de até 120 segundos [Liao et al. 2006]. Nesse ponto as redes P2P ainda perdem para as redes CDN, a exemplo da já citada rede Akamai que leva poucos segundos para iniciar a exibição do vídeo.

Após o tempo de recebimento dos primeiros dados e *buffering* do vídeo, começamos a verificar, a cada 8 segundos, a presença do pedaço necessário à reprodução segundo dois critérios: corte e pausa. Estes dois critérios representam os cenários encontrados tanto nos sistemas de TV convencionais (ausências de sinal causam cortes na transmissão), quanto na maioria das transmissões pela Internet (vídeos do YouTube pausam diante da ausência de dados).

Uma vez colhido os dados sobre as ausências para os 4 algoritmos e para o método tradicional do BitTorrent, nos voltamos para análise dos dados por meio da Regressão Linear e do K-Means.

4. Resultados Experimentais

Nesta seção, apresentamos e analisamos nossos resultados experimentais. Nossos objetivos são (a) mostrar o quão previsível é o comportamento dos algoritmos de seleção de pares com relação a ausência de pedaços no momento da reprodução, usando a Regressão Linear e (b) avaliar possíveis agrupamentos, "estágios" de qualidade, durante o experimento por meio do K-Means. Por uma questão de síntese vamos apresentar com maiores detalhes apenas os dados provenientes dos experimentos com ASAP e critério corte.

4.1. Ambiente de Experimentação

Os experimentos foram realizados no Planetlab, onde diversas máquinas foram selecionadas de acordo a disponibilidade e carga na hora do experimento (detalhes [de Sousa 2013]). Com o objetivo de comparar a qualidade da distribuição do vídeo, os cenários emulavam esse evento através de *swarms* Bittorent. As variações envolveram os algoritmo de seleção de pedaços e a quantidade de *leechers* no *swarm*.

Nosso experimento foi executado em grupos de 25, 50, 75, 100 e 125 *peers*, com um *tracker* e um *seeder*. Os dados estatísticos foram extraídos do arquivo "Azureus_stats.xml". O Vuze, por padrão, gera esse arquivo a cada 30 segundos, quando as estatísticas estão ativadas. Desse arquivo foram obtidos o *downloaded* e o *uploaded* de cada *peer* com o respectivo IP. Esse registro aconteceu durante 2.000 segundos, mesmo tempo utilizado no trabalho com o GoalBit [Bertinat et al. 2009].

Em [Liao et al. 2006] vemos que o tempo de vida médio dos pares é exponencialmente distribuído e no estudo de [de Faria 2010] podemos ver que a entrada dos pares também segue uma exponencial. Considerando essas informações, a chegada dos *leechers* no *swarm* foi projetada para seguir uma exponencial, distribuindo a entrada nos primeiros 200 segundos a uma taxa média de $\lambda = 0,05$ peer/s.

4.2. Avaliação dos Resultados Experimentais

A seguir, avaliamos os dados experimentais tendo em vista os objetivos expostos no início desta seção. Os dados a seguir são do experimento com 100 pares e foram convertidos

para índices, tanto o número de máquinas que tiveram perdas quanto o tempo do experimento. O índice dos pares é calculado pela divisão do “total de máquinas que registraram ausência de pedaço em determinado momento do experimento” pelo “número de pares (100) vezes o número de repetições (30)”. Já o índice do tempo divide “um determinado momento do experimento” pelo “tempo total do experimento (2000)”.

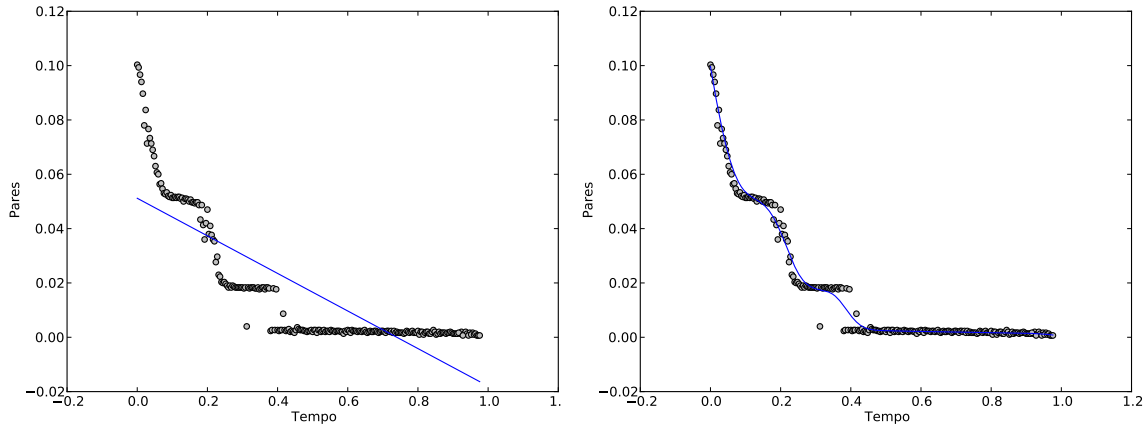

Figura 1. Regressão linear com Vuze sem *plugin*. A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

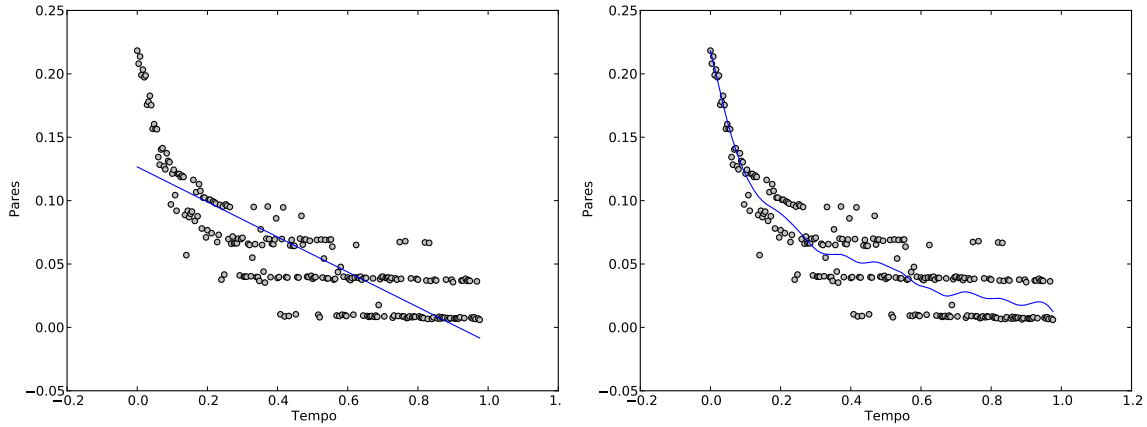

Figura 2. Regressão linear com Ono. A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

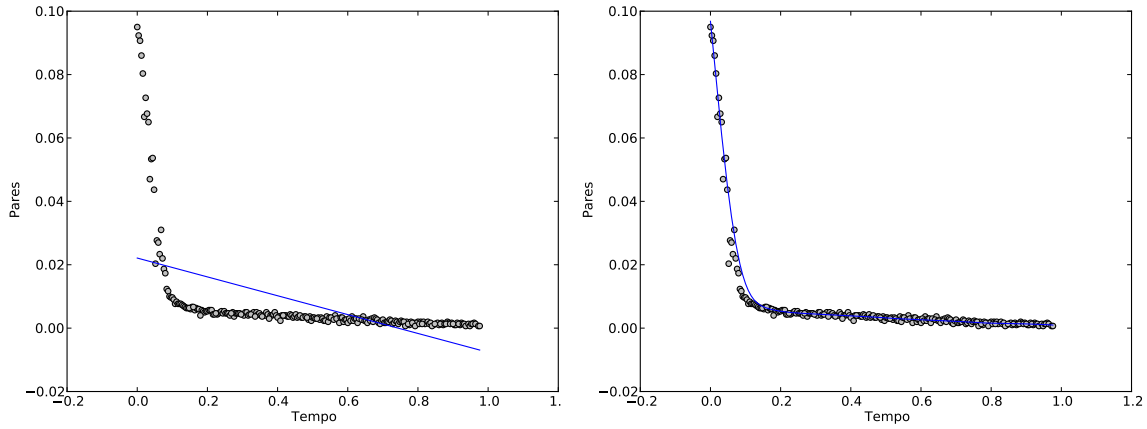

Figura 3. Regressão linear com P4P. A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

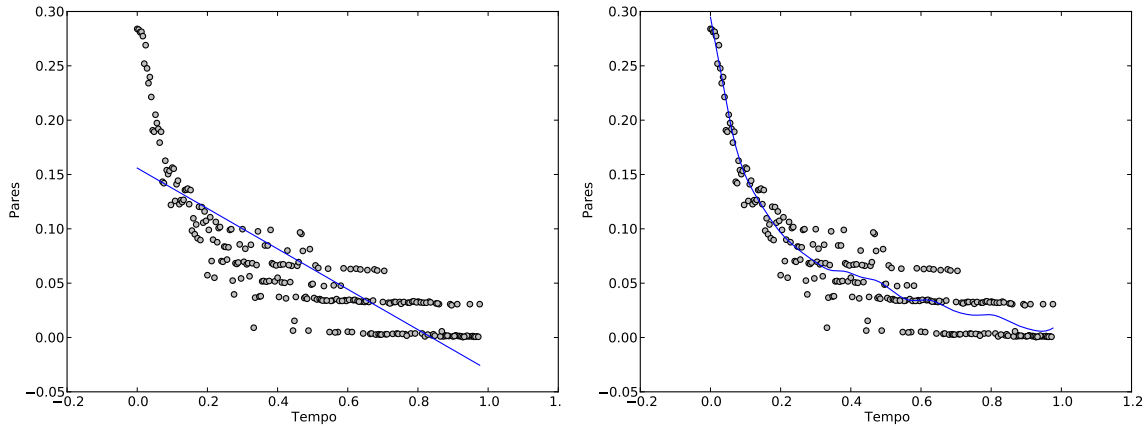

Figura 4. Regressão linear com Yukka. A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

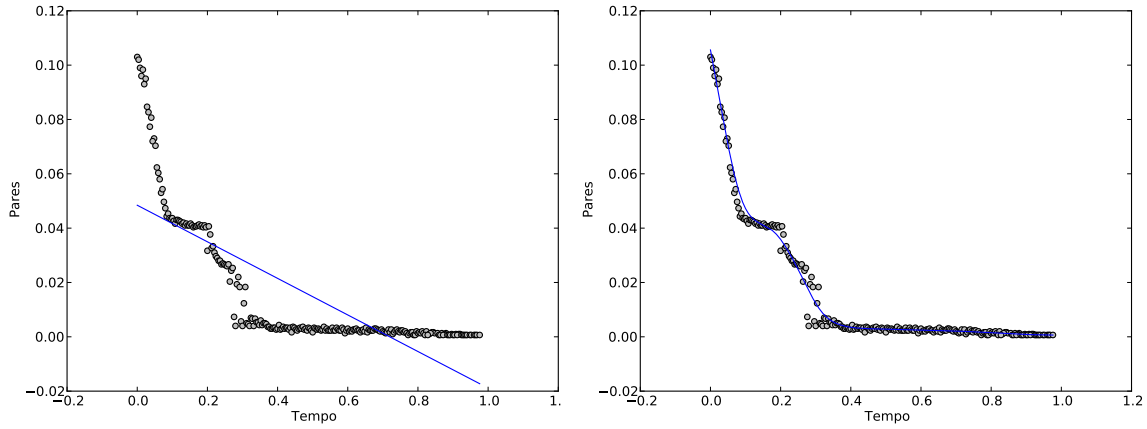

Figura 5. Regressão linear com S4Q. A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

Pela *best-fit line* à esquerda das Figuras 1, 2, 3, 4 e 5, podemos ver que, mesmo no caso do P4P, existe uma grande variação que pelo fato dos gráficos não apresentarem um comportamento linear. Porém, foi possível observar uma grande previsibilidade nas soluções P4P e S4Q, à direita das Figuras 3 e 5, ao utilizarmos *smoothing value* de k=0,03. O BitTorrent tradicional, apesar do método que utiliza, também apresentou uma baixa variação (veja Figura 1). Isso e o resultado do P4P não eram esperados. Nossa expectativa para esses dois algoritmos era algo mais parecido com as Figuras 2 e 4, devido ao uso do ASAP, pois a chegada desordenada dos pedaços deveria resultar no registro de várias ausências ao longo do experimento. Diante disso resolvemos consultar os *traces* do experimento com 75 pares, escolhemos um número menor de pares porque acreditamos que, quanto menor o grupo menor a eficiência do mesmo.

Na Figura 6 podemos ver que o "método tradicional" começa com várias ausências, seguidas por uma queda brusca. Não é exatamente o esperado, mas já mostra uma variação de comportamento que pode comprometer a previsibilidade. Já a Figura 7 nos mostra que o P4P pode ser totalmente imprevisível e inadequado para a transmissão de vídeo, pois o resultado é bem diferente do apresentado na Figura 3. Por fim, a confrontação entre as Figuras 5 e 8 mostra que o S4Q mantém o comportamento de reduzir as ausências, melhorando a experiência do usuário.

Em nossos experimentos o S4Q também utilizou o ASAP, porém ele consegue "induzir" a uma chegada sequencial dos pedaços sem interferir no ASAP. Dessa forma, o S4Q consegue um efeito parecido com o uso do ASSP, mas sem causar o problema da latência (o tempo médio de *download* com ASSP chega a dobrar em alguns cenários se comparado com ASAP).

Entre os algoritmos que apresentaram uma grande variação (como o Ono, Figura 2), podemos ver que os pontos parecem formar 4 grupos quase horizontais, por isso decidimos utilizar o K-Means para verificar essa impressão, ou seja, verificar a existência de grupos.

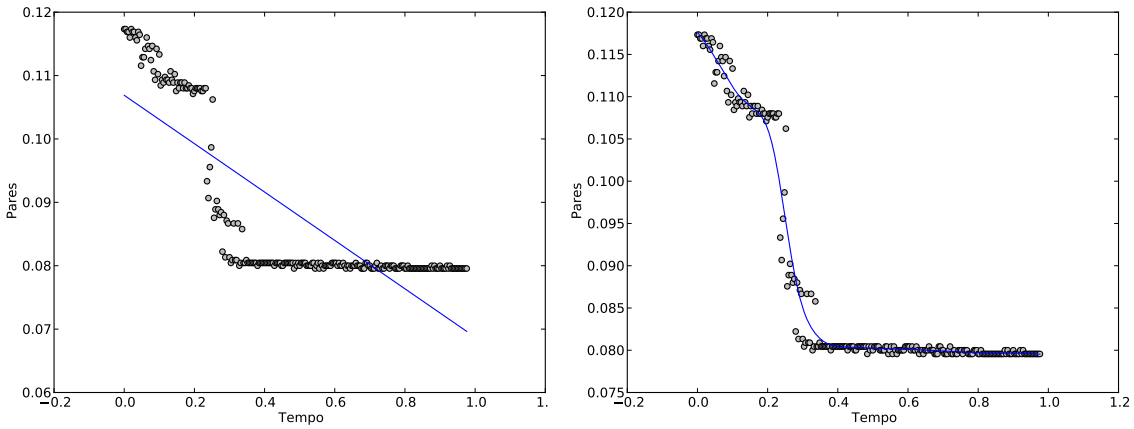

Figura 6. Regressão linear com Vuze sem *plugin* (75 Pares). A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

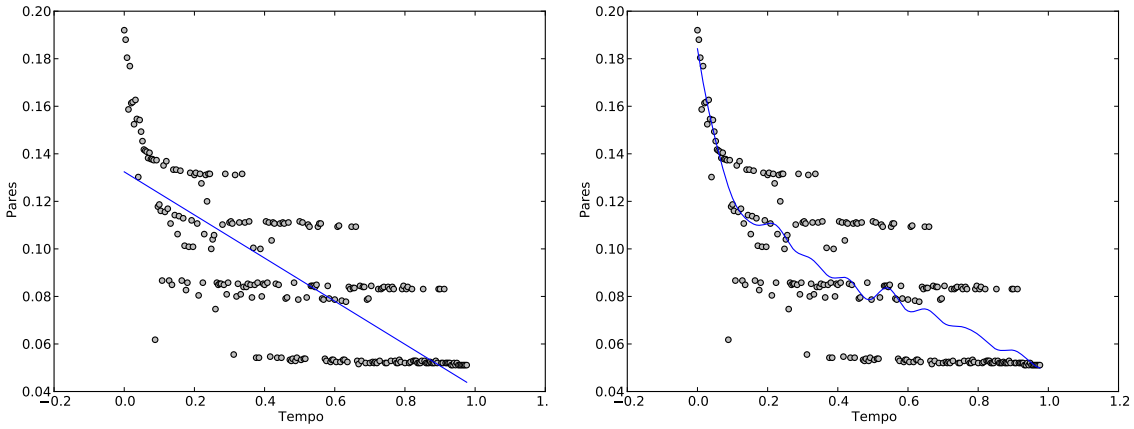

Figura 7. Regressão linear com P4P (75 Pares). A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

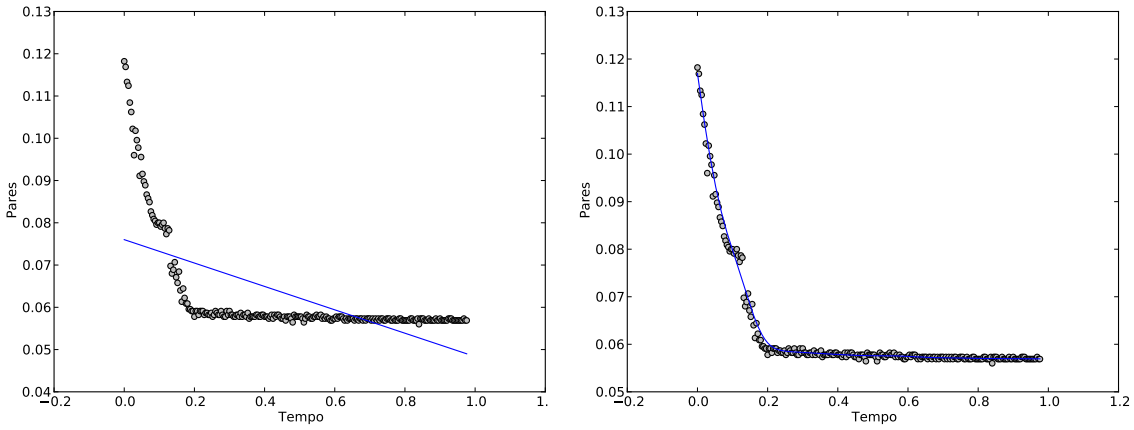

Figura 8. Regressão linear com S4Q (75 Pares). A esquerda vemos projeção da *best-fit line* e a direita do *smoothing value* de k=0,03

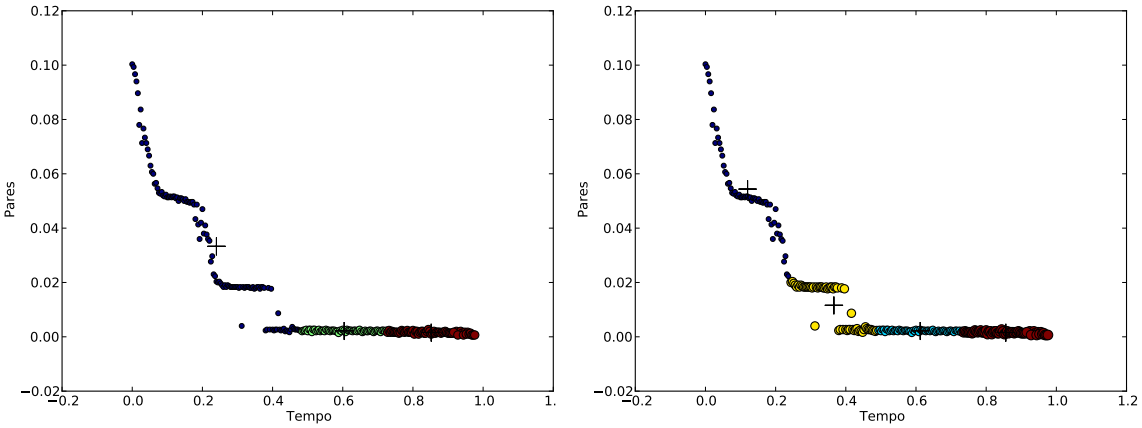

Figura 9. K-Means com Vuze sem *plugin*. A esquerda vemos projeção com 3 grupos e a direita com 4

Na Figura 10 os centróides até ficaram posicionados no que pareciam ser os grupos horizontais, mas a divisão não ocorreu como o esperado, resultando em grupos puramente divididos na vertical. A Figura 12, referente ao Yukka, não indica a existência de grupos verticais ou horizontais devido a forma como os pontos ficaram dispersos. Já o apresentado nas Figuras 9, 11 e 13 podem servir como indicadores de possíveis estágios da qualidade de recepção do vídeo durante sua transmissão. Ficando o Vuze e o S4Q com 4 estágios e o P4P com 3.

Em valores aproximados, o Vuze sem *plugin* e o S4Q começam com uma faixa que vai até 20%, depois de 20% à 50%, de 50% à 75% e, por fim, a última faixa que evolve o restante a partir de 75% do tempo do experimento.

O P4P começa com uma faixa que vai até 20%, depois de 20% à 50% e, finalmente, a última faixa que começa em 50% do tempo do experimento.

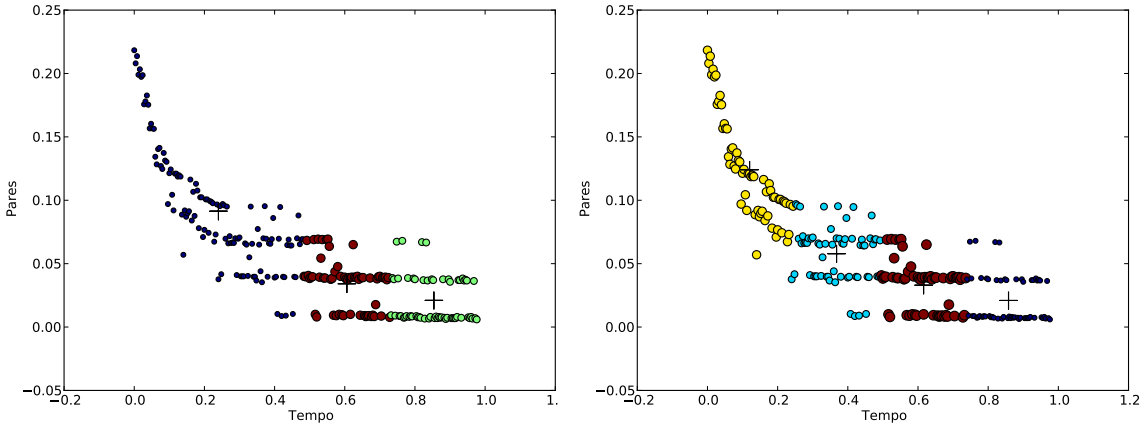

Figura 10. K-Means com Ono. A esquerda vemos projeção com 3 grupos e a direita com 4

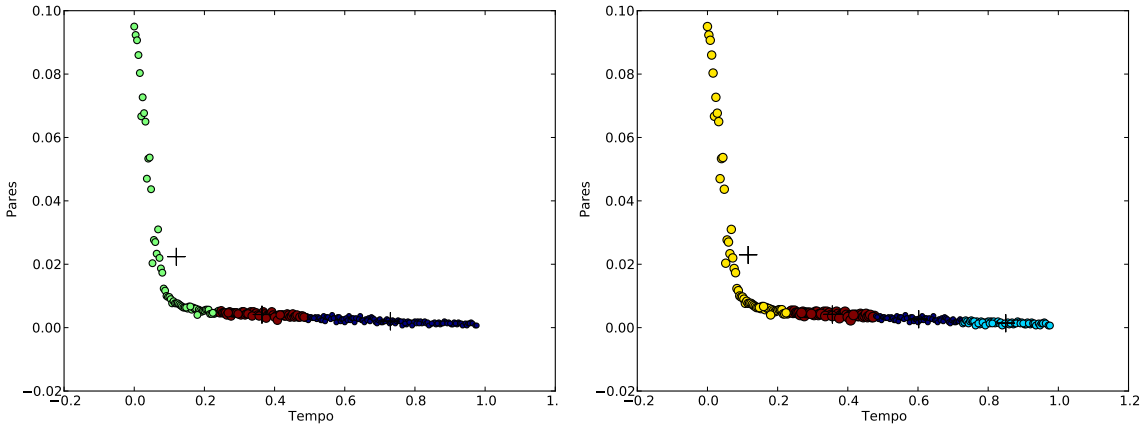

Figura 11. K-Means com P4P. A esquerda vemos projeção com 3 grupos e a direita com 4

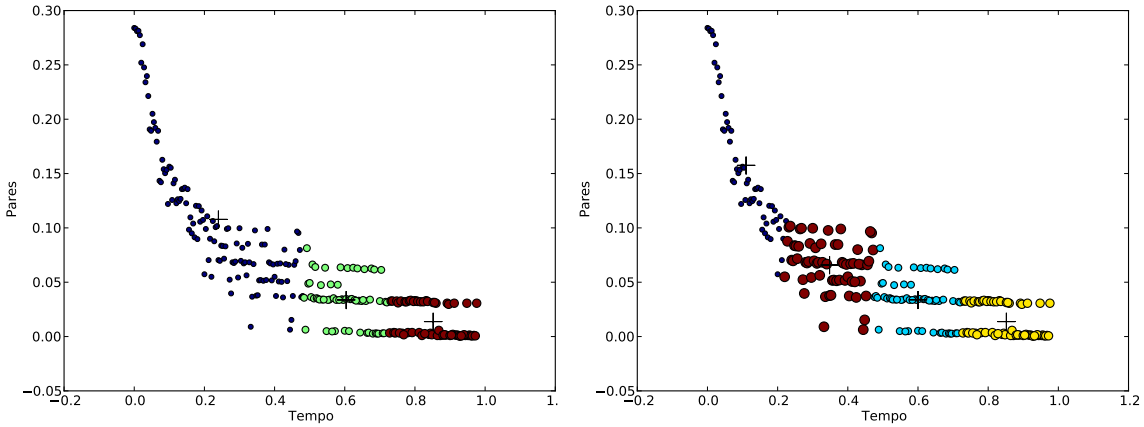

Figura 12. K-Means com Yukka. A esquerda vemos projeção com 3 grupos e a direita com 4

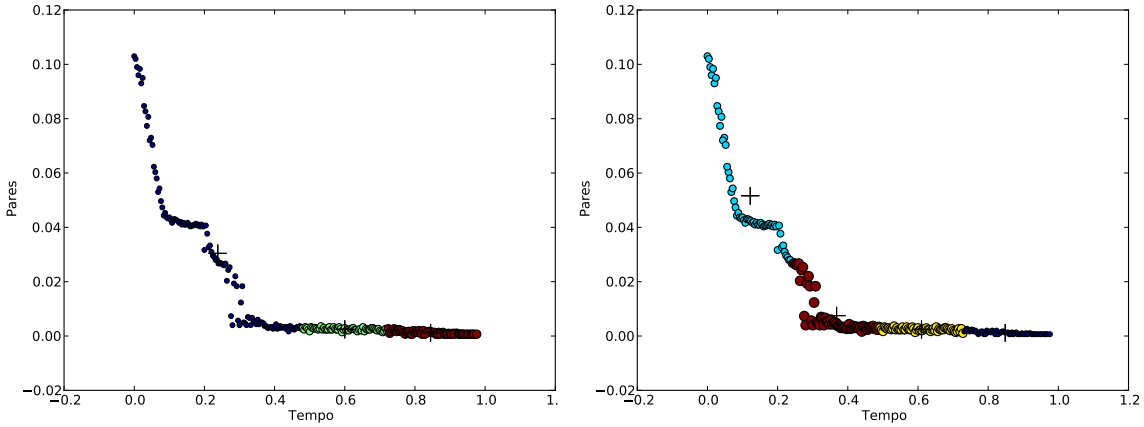

Figura 13. K-Means com S4Q. A esquerda vemos projeção com 3 grupos e a direita com 4

5. Conclusão

Este trabalho apresentou um comparativo entre 4 soluções para seleção de pares e o método tradicional do BitTorrent através de uma transmissão de vídeo para 100 pares e tomando por base a verificação de pedaços no critério corte e dessa forma avaliando as interrupções, durante a transmissão do vídeo pela rede.

A partir dos resultados obtidos podemos concluir que, (1) utilizando Regressão Linear, é possível prever as ausências de pedaços para o método tradicional do BitTorrent com baixa variação, mesmo que a essência de seu funcionamento seja praticamente aleatória, e com baixíssima variação a ausência de pedaços para o P4P e para o S4Q. Esse fato os torna mais atraentes para o uso na transmissão de vídeo pois é possível inferir previamente a qualidade da entrega no usuário final; (2) e que, utilizando o K-Means, podemos realizar agrupamentos que representam os estágios de qualidade durante a transmissão do vídeo, sendo 4 estágios para o Vuze o e S4Q, e 3 para o P4P.

Extendendo a pesquisa sobre os experimentos com 75 pares, vemos que o S4Q não apresenta grandes alterações na recepção dos pedaços. Logo, uma Regressão Linear nele, com k=0,03, proporciona uma pequena margem de erro tanto no experimento com 75 quanto no experimento com 100 pares. Além de mostrar que o S4Q tende a reduzir a ausência de pedaços ao longo do tempo, os resultados também sinalizaram que S4Q é mais previsível que os outros algoritmos e, portanto, melhor para transmissão de vídeo.

Nossa pesquisa mostrou pontos que precisam ser explorados com mais detalhes e que podem fazer com que os algoritmos para seleção de pares alcancem resultados ainda melhores, esses pontos passam pela implementação de métodos que sejam capazes de prever a QoE e atuar na sua melhora. No entanto, essas e outras extensões ficam para trabalhos futuros.

Referências

Bertinat, M. E., Vera, D. D., Padula, D., Amoza, F. R., Rodríguez-Bocca, P., Romero, P., and Rubino, G. (2009). Goalbit: The first free and open source peer-to-peer streaming network. *IEEE LANC*.

Choffnes, D. R. and Bustamante, F. E. (2008). Taming the torrent: A practical approach to reducing cross-isp traffic in peer-to-peer systems. *ACM SIGCOMM*.

de Faria, M. M. (2010). Lidando com tráfego egoísta em redes bittorrent. Master thesis, Centro de Ciências Exatas e Tecnológicas, Universidade Federal do Estado do Rio de Janeiro - UNIRIO.

de Sousa, P. R. (2013). Seleção de pares baseada em qoe para transmissão de vídeo em redes p2p bittorrent. Master thesis, Centro de Ciências Exatas e Tecnológicas, Universidade Federal do Estado do Rio de Janeiro - UNIRIO.

de Sousa, P. R., de Lucena, S. C., Diniz, M. C., de Aragão Rocha, A. A., and Menasché, D. S. (2014). S4q: Um algoritmo para seleção de vizinhos baseado em qoe para sistemas p2p de transmissão de vídeo. *WP2P+ da XXXII SBRC*.

Harrington, P. (2012). *Machine Learning in Action*. Manning Publications.

Krishnan, S. and Sitaraman, R. (2012). Video stream quality impacts viewer behavior: Inferring causality using quasi-experimental designs. *ACM IMC*.

Liao, X., Jin, H., Liu, Y., Ni, L. M., and Deng, D. (2006). Anysee: Peer-to-peer live streaming. *IEEE INFOCOM*.

Moraes, I. M., Campista, M. E. M., Moreira, M. D. D., Rubinstein, M. G., Costa, L. H. M. K., and Duarte, O. C. M. B. (2008). Distribuição de video sobre redes par-a-par: Arquiteturas, mecanismos e desafios. *Minicursos do XXVI SBRC*.

NetFlix (2013). Netflix: Que velocidade de internet banda larga preciso ter para assistir online? `https://signup.netflix.com/HowItWorks`. Acessado: Jan/2013.

Polaczyk, B. and Cholda, P. (2010). Bittorrent traffic localization via operator-related information. *IEEE ICC*.

Rodríguez-Bocca, P. (2008). *Quality-Centric Design of Peer-to-Peer Systems for Live-Video Broadcasting*. Ph.d. dissertation, l'Université de Rennes.

Vuze (2013). Vuze: Plugin development guide. `http://wiki.vuze.com/w/Plugin_Development_Guide`. Acessado: Jan/2013.

Xie, H., Yang, Y. R., Krishnamurthy, A., Liu, Y., and Silberschatz, A. (2008). P4p: Provider portal for applications. *ACM SIGCOMM*.