O Problema de Deteção e Localização de Eventos em Séries Temporais Aplicado a Redes de Computadores

Diego Ximenes[1], **Gabriel Mendonça**[1], **Gustavo H. A. Santos**[1], **Edmundo de Souza e Silva**[1], **Rosa M.M. Leão**[1], **Daniel S. Menasché**[1]

[1]Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro, RJ.

{diego,gabriel,gustavo,edmundo,rosam,sadoc}@land.ufrj.br

Abstract. *Nowadays, ISPs provide the infrastructure for essencial services. Therefore, detecting and localizing network problems automatically is an important issue. Despite current monitoring efforts, ISPs still do not have appropriate tools for coping with the non-trivial task of processing an enourmous amount of data generated by QoS measurements performed dozens of times per day. We propose an approach for detecting and localizing network problems from timeseries resulting from measuremnts and traceroutes. Our data analysis framework is based on the so called "change point detection problem", unsupervised learning and network topology inference.*

Resumo. *Atualmente provedores de serviço de Internet (ISPs) fornecem infraestrutura para diversos serviços essenciais. Portanto, é fundamental detetar e localizar problemas na rede de forma automática. Apesar de esforços de monitoramento feito pelos ISPs, eles ainda não têm as ferramentas apropriadas para processar a enorme quantidade de dado gerado por medições de QoS realizadas dezenas de vezes por dia. Essa tarefa não é trivial. Propomos uma abordagem para detetar e localizar automaticamente problemas nas redes dos ISPs a partir de séries temporais resultantes de medições de QoS e* traceroutes*. Nosso* framework *é baseado no problema denominado de* change point detection*, em classificação não-supervisionada e inferência de topologia de rede. Aplicamos nosso framework em dados reais obtidos durante vários meses.*

1. Introdução

Provedores de serviço de Internet (ISPs) prestam um serviço de infraestrutura básica. Pelo fato da Internet prover uma infraestrutura crítica é fundamental detetar automaticamente problemas nas redes dos ISPs. Entretanto, as ferramentas disponíveis ainda não permitem uma análise precisa e automática de deteção de problemas. Na medida em que a complexidade e a capilaridade da topologia dos ISPs aumenta, os problemas de deteção, localização e correção de falhas tendem a se agravar.

É possível monitorar a qualidade de serviço através de medições realizadas em casas de usuários voluntários. As métricas de QoS podem ser obtidas a partir de *software* embarcado em roteadores residenciais conectados diretamente ao modem ou de computadores, celulares ou outros equipamentos ligados à rede doméstica. Este tipo de *software* pode realizar medições periódicas entre os roteadores residenciais e servidores localizados em pontos estratégicos da operadora. Neste caso, obtém-se milhões de amostras, cuja análise não é trivial. Por exemplo, para fins de detetar problemas na rede, a identificação

visual de mudanças de padrões nas séries temporais é desafiadora, requerendo soluções automatizadas. A deteção de mudança de padrão estatístico em uma série temporal é denominada na literatura de *change point detection problem*, as soluções para esses problemas devem ser capazes de lidar com séries temporais não estacionárias, a fim de identificar *change points* que sinalizem potenciais problemas.

A deteção automatizada de *change points*, em geral, pode se beneficiar de dados históricos sobre eventos reportados pelos administradores de rede. Neste caso tais eventos poderiam, por exemplo, indicar o início de uma mudança de comportamento de uma medida de QoS. Entretanto, um *dataset* que aponte a falha e estabeleça a correlação com as medidas fim a fim, isto é, que possibilite obter o *ground truth* dos eventos de rede, não é simples e em geral não existe. Assim, caracteriza-se como um desafio adicional a necessidade de uma abordagem não-supervisionada para a solução do problema.

Este trabalho propõe um *framework* baseado em medições fim-a-fim em redes reais e *traceroutes* para a deteção e localização de eventos de rede de maneira automática. Um evento de rede pode ter várias causas raiz como, por exemplo: um congestionamento transitório em uma ou mais regiões da rede; mudanças de rotas; alguma mudança no funcionamento de um equipamento como uma falha parcial, intermitente ou total em um roteador. Em resumo, alguma mudança que venha a afetar a qualidade do serviço percebida pelo usuário final. O procedimento de localização de eventos define um conjunto de possíveis pontos onde o evento pode ter iniciado ou terminado.

A principal contribuição deste trabalho é a elaboração de um *framework* de análise de dados capaz de detetar e localizar eventos de maneira automática. A deteção de eventos é realizada a partir da identificação de mudanças estatísticas nas séries temporais de medições **reais** de QoS de diferentes clientes. A localização dos eventos é feita a partir da correlação espaço-temporal destas mudanças considerando a topologia da rede estimada a partir das medições de *traceroute*. É importante ressaltar que nosso objetivo é a localização de eventos, isto é determinar mudanças estatísticas de QoS que tenham afetado um ou mais usuários, e as regiões prováveis que possam explicar mudanças de QoS medidas a partir dos roteadores domésticos. Não é nosso propósito a deteção das causas raiz desses eventos. O *framework* foi aplicado em dados reais e, na falta de um *dataset* de causas raiz, foi aplicada uma análise não supervisionada.

O artigo é organizado da seguinte forma. A seção 2 aborda o problema de *change point detection*. A seção 3 explica o funcionamento do *framework* de análise de dados utilizado para deteção e localização de eventos. A seção 4 mostra os resultados obtidos utilizando a técnica proposta. A revisão da literatura é apresentada na seção 5. Por fim, a seção 6 apresenta a conclusão e trabalhos futuros.

2. *Change Point Detection*

Algoritmos de *change point detection* têm por objetivo identificar pontos no tempo em que ocorrem alterações em propriedades estatísticas de uma série temporal, sem levar em consideração um comportamento padrão pré-definido. A literatura sobre *change point detection* é bastante vasta, sendo possível a utilização de diferentes técnicas para a identificação das mudanças estatísticas nas séries temporais, como janelas deslizantes [Kifer et al. 2004], cadeias de Markov ocultas [Kehagias 2004, Luong et al. 2013, Montañez et al. 2015] e inferência bayesiana [Fearnhead 2006,

Adams and MacKay 2007]. A técnica utilizada neste trabalho foi aplicada a partir do processo de otimização definido em [Maidstone et al. 2017]. Uma visão geral sobre o problema de *change point detection* pode ser encontrada em [Truong et al. 2018, Aminikhanghahi and Cook 2017].

Na literatura, um *change point* é um ponto em que o modelo usado para capturar características da série temporal precisa ser alterado devido a mudanças na série tal que o modelo em uso não mais descreve o comportamento observado de maneira acurada. Não é simples detetar mudanças estatísticas. Considere, por exemplo, que os valores de uma série temporal observados até um ponto x_{t^*} sigam uma distribuição gaussiana $N(\mu, \sigma_a)$ e, após o ponto x_{t^*} a variância da série temporal mude, de forma que a distribuição que mais se adéque à descrição da série seja uma nova gaussiana $N(\mu, \sigma_b)$. Neste exemplo, x_{t^*} representa o ponto a partir do qual a distribuição muda, ou seja, x_{t^*} é um *change point*. Um *outlier*, por outro lado, é um ponto ou um conjunto de pontos que fogem do comportamento padrão da série temporal, mas cujo comportamento pode ainda ser explicado pelo modelo vigente com boa precisão. Para exemplificar brevemente, suponha que em uma série temporal modelada por uma gaussiana $N(\mu, \sigma)$ o ponto x_{t_i} tenha uma probabilidade baixa segundo o modelo, mas que os pontos seguintes $(x_{t_{i+1}}, x_{t_{i+2}}, \ldots, x_{t_{i+n}})$ tenham alta probabilidade de serem gerados pela gaussiana $N(\mu, \sigma)$. Desta forma, embora um *outlier* possa ter uma probabilidade baixa de ser gerado, ainda assim ele é previsto pelo modelo considerado, isto é, o evento tem uma probabilidade não nula de ocorrer, e a sua existência não significa uma mudança na distribuição. Em problemas característicos de deteção de anomalia, após a ocorrência do *outlier*, a série temporal volta ao seu comportamento original, o que não ocorre quando existe um *change point* [Su et al. 2013]. Através desse exemplo, não é difícil notar os problemas para se detetar mudanças estatísticas: qual o intervalo de tempo para considerar que houve mudanças na série temporal? como detetar e descartar *outliers*? como lidar com dados incompletos? quantas amostrar são necessárias para ajustar os parâmetros do algoritmo usado?

Algoritmos de deteção de anomalia assumem que uma série temporal possui um comportamento padrão e localizam pontos que desviam deste padrão (*outliers*). O sistema Argus [Yan et al. 2012] é um exemplo de aplicação de deteção de anomalia para a identificação e localização de problemas de rede. No entanto, foi percebido nos dados coletados que mudanças na distribuição de probabilidade de métricas de qualidade de rede persistem por períodos relativamente longos de tempo. Portanto, a abordagem de *change point detection* é mais adequada para deteção de potenciais problemas. As Figuras 9(c,d) são exemplos de mudança de comportamento de séries temporais que persistem por um longo período de tempo após o *change point* detetado. Note também o comportamento periódico da série temporal presente na Figura 9(d), o que torna o problema mais difícil.

Neste trabalho, para detetar um *change point*, consideramos alterações na distribuição de probabilidade das amostras. *Outliers* não são considerados como mudanças estatísticas. O número de *change points* é desconhecido e o número de amostras dentro dos intervalos definidos por estes pontos também não é fixo. Utilizamos séries temporais univariadas de diferentes tamanhos e intervalo entre amostras não uniforme. Existem diferentes conjuntos de séries, cada uma correspondendo a uma métrica de QoS obtida de medições realizadas a partir de roteadores residenciais (por exemplo, roteadores tipo TP link) de clientes de uma rede real.

2.1. Notação

Uma série temporal composta de n pontos é definida por dois vetores, $\mathbf{x} = (x_1, \ldots, x_n)$ e $\mathbf{y} = (y_1, \ldots, y_n)$, onde y_i indica o valor da i-ésima amostra e x_i indica o tempo associado à i-ésima amostra. Os pontos são ordenados no tempo, ou seja, $x_{i-1} < x_i$, $i = 2, \ldots, n$. Séries temporais com intervalos não-uniformes são consideradas neste trabalho. Logo, o intervalo entre amostras $x_i - x_{i-1}$ pode ser distinto para diferentes valores de i. Um subconjunto de amostras é definido através da notação $\mathbf{y}_{s:t} = (y_s, \ldots, y_t)$, $s \leq t$.

A existência de k *change points* implica na separação dos dados em $k+1$ segmentos, também conhecidos como janelas. Seja τ_i o i-ésimo *change point* para $i = 1, \ldots, k$, com $\tau_0 = 0$ e $\tau_{k+1} = n$. Desta forma, o i-ésimo segmento é definido por $\mathbf{y}_{\tau_{i-1}+1:\tau_i}$, assumindo que $\tau_{i-1} < \tau_i$ para $i = 1, \ldots, k+1$.

Utilizando esta notação, dada uma série temporal um algoritmo de *change point detection* tem como objetivo a obtenção do número de segmentos e dos *change points*, ou seja, a obtenção de k e $\boldsymbol{\tau} = (\tau_0, \tau_1, \ldots, \tau_{k+1})$.

2.2. Algoritmo de *change point detection*

Diferentes algoritmos de *change point detection* são encontrados na literatura, e cada um pode ser mais adequado para lidar com um determinado tipo de problema. Considerando as características das séries temporais de medições em redes, escolhemos na literatura um algoritmo de *change point detection* baseado em um modelo de otimização definido em [Maidstone et al. 2017]. Essa abordagem se mostrou adequada no estudo que realizamos, pois foi capaz, por exemplo, de eliminar adequadamente outliers, foi computacionalmente eficiente para lidar com o grande número de séries de que dispomos, etc. (Esse é um estudo preliminar e pretendemos continuar avaliando diferentes métodos.)

Na abordagem de [Maidstone et al. 2017] , o problema é definido a partir de uma função de custo que mede a "homogeneidade"de cada janela. O processo de otimização é responsável pela escolha dos pontos de mudança que otimizem de maneira global a função de custo.

O número de pontos de mudança nos dados de medição não é conhecido previamente. Desta forma, o problema é definido a partir do caso penalizado [Maidstone et al. 2017]. A função de otimização é definida pela função de custo de cada segmento $C(\mathbf{y}_{\tau_{i-1}+1:\tau_i})$ e por uma função não-decrescente de penalidade $g(k)$, sendo o valor ótimo obtido através de programação dinâmica. A função a ser otimizada é dada por:

$$\min_{k, \boldsymbol{\tau}_{1:k}} \sum_{i=1}^{k+1} C(\mathbf{y}_{\tau_{i-1}+1:\tau_i}) + g(k) \tag{1}$$

Diversas funções de custo C e penalidade $g(k)$ podem ser utilizadas pelo processo de otimização. Escolhas comuns para a função de custo incluem o erro quadrático médio e o *negative maximum log-likelihood* de uma distribuição de probabilidade.

3. *Framework*

A Figura 1 ilustra os processos que compõem nosso *framework* para deteção e localização de eventos em redes de computadores. Dado um conjunto de séries temporais de medições

de QoS de diversos clientes em uma rede, a análise consiste na execução dos seguintes passos:

1. **Filtragem de clientes**: Remoção de séries com problemas, por exemplo, aquelas com um número pequeno de amostras de medição.
2. **Avaliação de correlação temporal (via *change point detection*)**: Processamento das séries temporais de medições de cada cliente para identificar *change points*, denominados *eventos de clientes*. Classificação dos pontos em eventos do tipo (1) ”piora”, no caso de piora da métrica de QoS; (2) ”melhora”, caso contrário.
3. **Avaliação de correlação espacial (via clusterização)**: Agrupamento de clientes de acordo com sua posição na rede, permitindo a posterior localização dos eventos. Chamaremos de *grupo* clientes agrupados em um mesmo *cluster* (vide Seção 3.1).
4. **Avaliação de correlação espaço-temporal**: Agrupamento de *eventos de clientes* de um mesmo grupo que sejam do mesmo tipo (piora / melhora) e tenham ocorrido aproximadamente ao mesmo tempo, identificando os *eventos de rede* (vide Seção 3.2).
5. **Análise de possível causa raiz (*root cause analysis*)**: Mapeamento dos eventos de rede com a estrutura de *grupos de clientes*, visando localizar o equipamento ou grupo de equipamentos possíveis causadores do evento (vide Seção 3.3).

A avaliação de correlação temporal foi discutida na seção 2.2. Detalharemos os processos seguintes nas próximas seções.

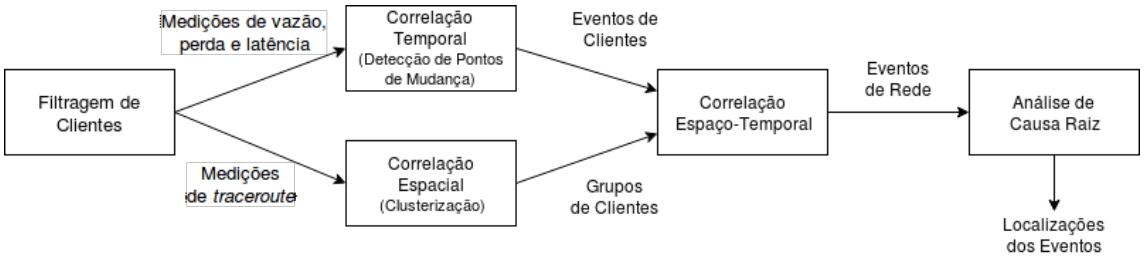


Figura 1. Visão geral dos processos que compõem o *framework* para deteção e localização de eventos em redes.

3.1. Correlação Espacial

O objetivo deste processo é agrupar clientes de acordo com sua localização na rede. Um *grupo de clientes* é formado por todos aqueles cujas rotas até o servidor de medição compartilham um determinado nó, sendo que um nó pode conter um ou mais equipamentos de rede. O processo de correlação espacial deve definir uma estrutura hierárquica de *grupos de usuários*, como exemplifica a Figura 2. A árvore criada nesta etapa será usada durante o processo de correlação espaço-temporal.

Quando a topologia do ISP é conhecida, essa tarefa torna-se mais simples. Entretanto, quando não temos acesso a essa informação podemos tentar reconstruir a estrutura da rede a partir de *traceroutes*.

A Figura 2 ilustra um exemplo de topologia real reconstruída com base em medições de *traceroute* feitas a partir dos gateways dos clientes. Os nós foram anonimizados. As arestas são orientadas dos clientes para o servidor de medição. Nesta árvore orientada, os nós com grau de entrada zero indicam equipamentos que aparecem apenas no primeiro salto (*hop*) do *traceroute* dos clientes.

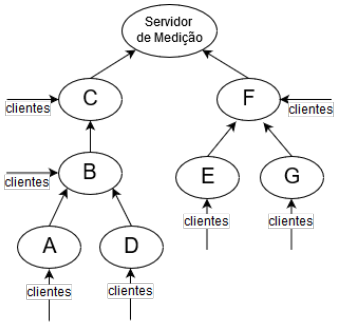


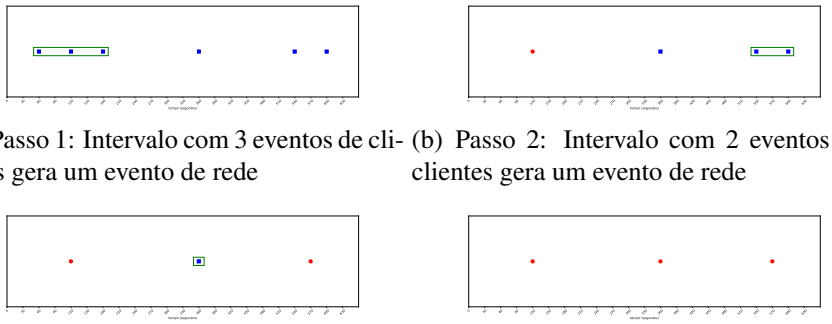
Figura 2. Exemplo de topologia reconstruída a partir de *traceroutes*.

3.2. Correlação Espaço-Temporal

A etapa de Correlação Espaço-Temporal tem como objetivo inferir *eventos de rede* que expliquem os pontos de mudança *change points* na etapa de Correlação Temporal e identificados para um mesmo *grupo de clientes* (obtido na Correlação Espacial). Exemplos de *eventos de rede* incluem a falha de um equipamento da rede, a configuração incorreta de um roteador, um congestionamento, etc.

Um mesmo *evento de rede* pode ser detetado em momentos distintos por clientes de um mesmo grupo, por exemplo, por conta do intervalo não-uniforme entre amostras das séries temporais. Portanto, é necessário elaborar um algoritmo para agrupar *eventos de clientes* de um mesmo grupo que aconteçam em um intervalo de tempo próximo.

Para resolver esse problema, criamos uma heurística para agrupamento de *eventos de clientes* detetados em um mesmo intervalo de tempo. O algoritmo proposto, denominado Votação Inexata Múltipla em Espaço Totalmente Ordenado, é inspirado no problema de Votação Inexata em Espaço Totalmente Ordenado [Parhami 1994]. Ilustramos a execução de nossa heurística na Figura 3. A cada passo do algoritmo, criamos um grupo de eventos a partir do intervalo de tempo — com tamanho máximo 2δ — que possui o maior número de *eventos de clientes* (*change points*) detetados. Para cada grupo, definimos um *evento de rede* com tempo dado pelo centro do intervalo. O algoritmo prossegue até que todos os eventos tenham sido agrupados.



(a) Passo 1: Intervalo com 3 eventos de clientes gera um evento de rede

(b) Passo 2: Intervalo com 2 eventos de clientes gera um evento de rede

(c) Passo 3: Intervalo com 1 evento de cliente gera um evento de rede

(d) Resultado final: 3 eventos de rede detetados

Figura 3. Exemplo de execução do algoritmo de Votação Inexata Múltipla em Espaço Totalmente Ordenado.

3.3. Análise de Possível Causa Raiz

A etapa de Análise de Causa Raiz permite a identificação de um conjunto de possíveis localizações para cada *evento de rede*, possibilitando, por exemplo, a localização de equipamentos com problema.

Para localizar eventos, precisamos relacionar *eventos de rede* de diferentes *grupos de clientes* buscando uma causa comum. Assumimos que, se um evento ocorre em um grupo, todo o tráfego do agrupamento correspondente é afetado. Essa suposição é razoável já que cada *grupo de clientes* compartilha um mesmo conjunto de equipamentos de rede. Assumimos também que o processo de correlação espacial é capaz de produzir uma estrutura hierárquica de grupos com no máximo uma aresta entre dois equipamentos.

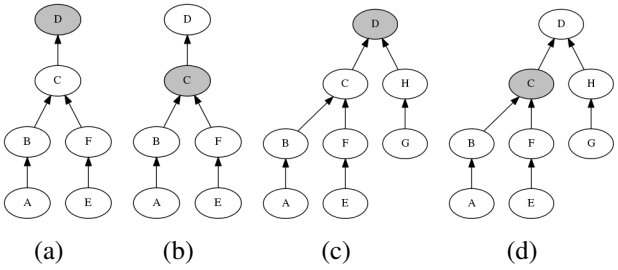


Figura 4. Exemplos de localização de *eventos de rede*.

Mostramos na Figura 4 alguns exemplos de topologia com a localização do *evento de rede* marcada em cinza. O procedimento para localização de *eventos de rede* se inicia nos vértices de grau de entrada zero ($\{$A, E, G$\}$ nos exemplos). Se uma fração pequena (parametrizável) dos clientes de um nó deteta um evento, assumimos que esse nó não é a causa do problema. Nesse caso, inferimos que os eventos detetados foram causados por um equipamento localizado antes do primeiro salto, afetando apenas alguns clientes.

Por outro lado, caso muitos clientes de um nó com grau de entrada zero detetem o evento, são possíveis três cenários: (1) o nó é a causa do problema; (2) os clientes compartilham um equipamento anterior ao nó que explica o problema; ou (3) o responsável pelo evento é um nó posterior. Quando não há informação sobre a topologia abaixo da camada IP (como no caso em que inferimos a estrutura da rede a partir de *traceroutes*), os casos (1) e (2) são indistinguíveis. No caso (3), devemos avaliar os demais nós no caminho até o servidor. Se há vários clientes ligados ao nó seguinte que não detetaram o evento, o problema certamente se encaixa nos casos (1) ou (2). Caso contrário, o procedimento é reexecutado para o nó seguinte.

De modo geral, se caminhos distintos detetam o mesmo *evento de rede*, então eles devem compartilhar ao menos um nó. Os nós que não são comuns podem ser descartados da lista de possíveis localizações.

Retornando aos exemplos da Figura 4 vemos que na Figura 4(a) um evento detetado em um subconjunto de clientes do nó A resulta na lista de possíveis locais com problemas: $\{$A, B, C, D$\}$. Um evento detetado por clientes de grau zero conectados ao nó E, implica nos possíveis locais com problema: $\{$E, F, C, D$\}$. Correlacionando as duas listas, temos $\{$C, D$\}$ como possíveis causas do evento. A Figura 4(b) mostra um cenário distinto (o evento ocorre em C) e o algoritmo chega à mesma conclusão anterior. Na Figura 4(c), a lista de possíveis locais construída a partir de cada nó com grau de entrada 1

mostra apenas D como causa do problema. Já na Figura 4(d), a análise encontra {A, B, C, D} partindo de A e {E, F, C, D} partindo de E. Como o evento não é detetado pelos clientes em G, C é a única localização possível para o problema.

4. Resultados

4.1. Dataset

A Figura 4.1 exemplifica a infraestrutura de medição a partir da qual os dados foram obtidos.

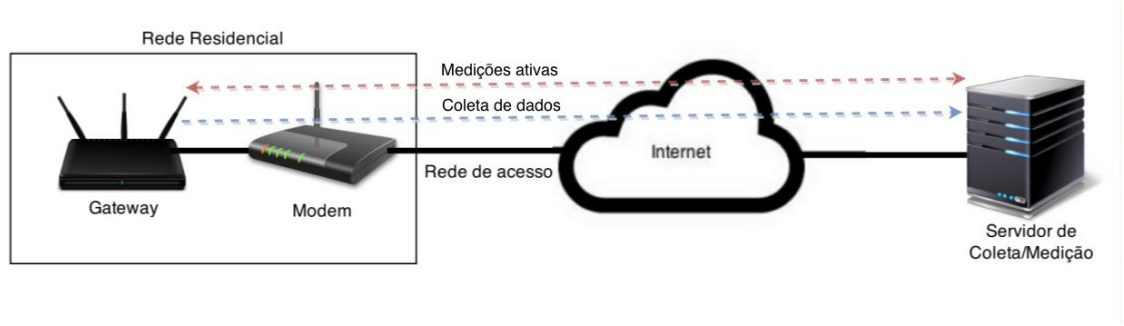


Figura 5. Infraestrutura de medição.

As medições de QoS que geraram as séries temporais consideradas foram realizadas durante 7 meses, entre maio e novembro de 2016 com mais de 2000 clientes voluntários. Foram consideradas como métricas de QoS a latência, a taxa de perda e a vazão *upstream*.

Os dados foram separados em *batches* de 10 dias. Aplicamos o algoritmo de *change point detection* para cada *batch*. As medições incluem 35 servidores espalhados em diversos estados do Brasil. O número médio de pares cliente-servidor com ao menos uma medição em cada *batch* é de 2246, sendo consideradas em média 741 séries temporais após a filtragem de usuários finais definida na seção 3.

4.2. Metodologia

Para reduzir o ruído, as séries temporais são pré-processadas antes da aplicação do algoritmo de *change point detection*. Utilizamos um filtro de medianas com janelas deslizantes centradas de tamanho w. Nesse filtro, o valor de y_i é dado pela mediana das amostras com índices entre $[i - w, i + w]$. O valor de w foi definido para cada métrica de rede após inspeção visual. Para medições de latência, a janela foi definida como $w = 13$. Para medições de vazão e perda, a janela foi definida como $w = 5$.

Os *change points* foram detetados a partir do algoritmo de otimização descrito na seção 2.2, que é aplicado considerando a série temporal completa. A falta de um *dataset* de eventos, isto é, de um dataset que indique instantes de tempo onde existiram problemas na rede, impossibilita uma análise supervisionada, de forma que a escolha dos parâmetros utilizados foi realizada a partir de inspeção visual. Utilizamos como função de custo C o *negative maximum log-likelihood* de uma Gaussiana. A função de penalidade varia de acordo com a métrica considerada. Para séries temporais de latência, $g(k) = 100k^2$. Para as séries temporais de perda e vazão, $g(k) = 200k$. Quando todas as amostras em uma

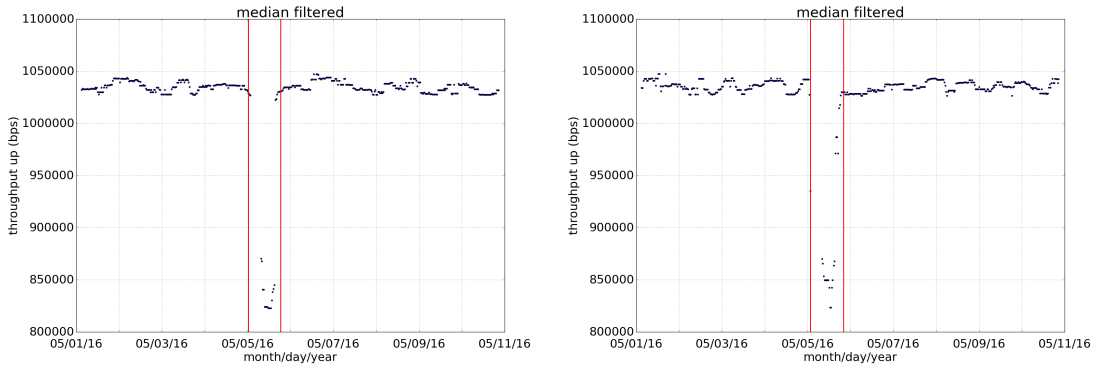
janela são iguais (levando a distribuições degeneradas de variância zero), adotamos um custo C constante, $C = \ln(0.000001)$.

O parâmetro δ usado na etapa de Correlação Temporal foi definido como 4 horas. Na Correlação Espaço-Temporal, consideramos que um *evento de rede* está associado a um *grupo de clientes* se ao menos 75% dos clientes do grupo geraram um *evento de cliente* na janela de tempo correspondente. Assumimos nas análises que qualquer mudança detetada por múltiplos clientes de uma mesma região em uma mesma janela de tempo têm como causa um único evento.

4.3. Eventos detetados

Nesta seção são apresentados alguns eventos detetados automaticamente pelo *framework* proposto. As séries temporais que serão mostradas foram escolhidas de forma a facilitar a identificação visual do leitor.

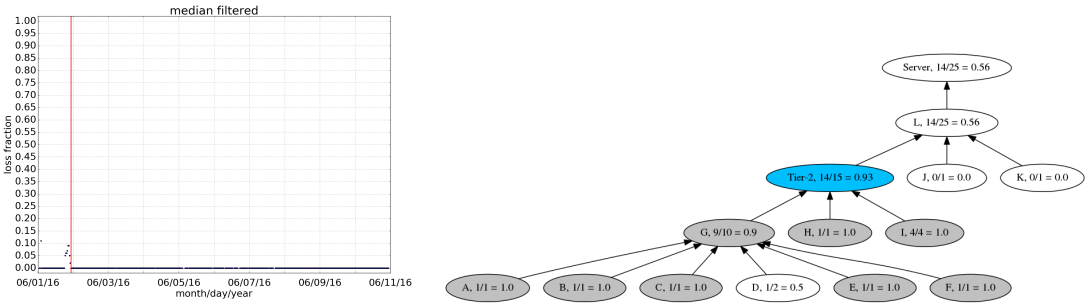
A Figura 6 mostra a série temporal de vazão *upstream* de dois clientes que pertencem a um mesmo vértice com grau de entrada igual a zero. Percebe-se que um evento de rede aconteceu na janela de tempo entre as linhas verticais. O vértice considerado agrupa 4 usuários diferentes e apenas os 2 usuários mostrados na figura identificaram o evento, o que indica que estes clientes compartilham um equipamento físico presente antes do primeiro *hop* identificado.



(a) Cliente 1 (b) Cliente 2

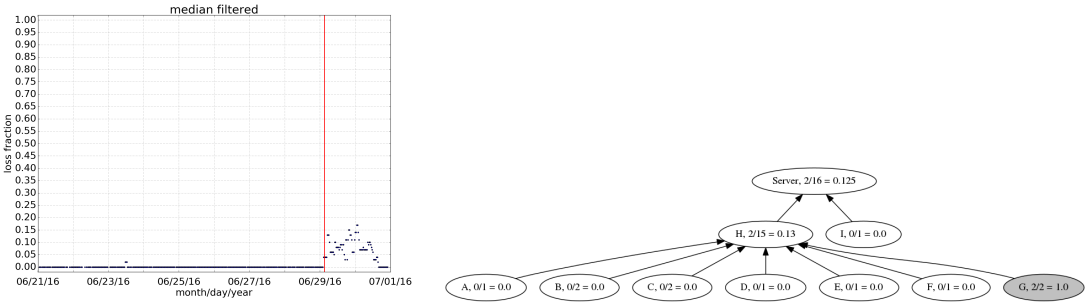
Figura 6. Evento identificado entre clientes do mesmo grupo

A Figura 7(a) mostra um evento identificado por um cliente que pertence ao vértice azul da Figura 7(b) (A Figura 7(a) aparentemente indica que o intervalo de mudança de comportamento da série seja pequeno. Entretanto esse é um problema de escala do gráfico, o intervalo corresponde a várias horas.) Cada vértice representado na figura é definido por uma tupla, onde o primeiro campo corresponde a um rótulo associado ao grupo de usuários e o segundo campo especifica a fração de clientes que detetaram o evento considerado. Os nós cinzas correspondem a possíveis locais de ocorrência do evento segundo a análise iniciada nos vértices de grau de entrada zero, sendo o nó azul identificado como o local onde o evento ocorreu após a correlação entre os resultados. Todos os nós que identificaram o evento mostraram comportamento semelhante ao da Figura 7(a).



(a) Cliente 1 (b) Grupo de usuários identificado

Figura 7. Evento localizado a partir da correlação espaço-temporal de múltiplos grupos de clientes



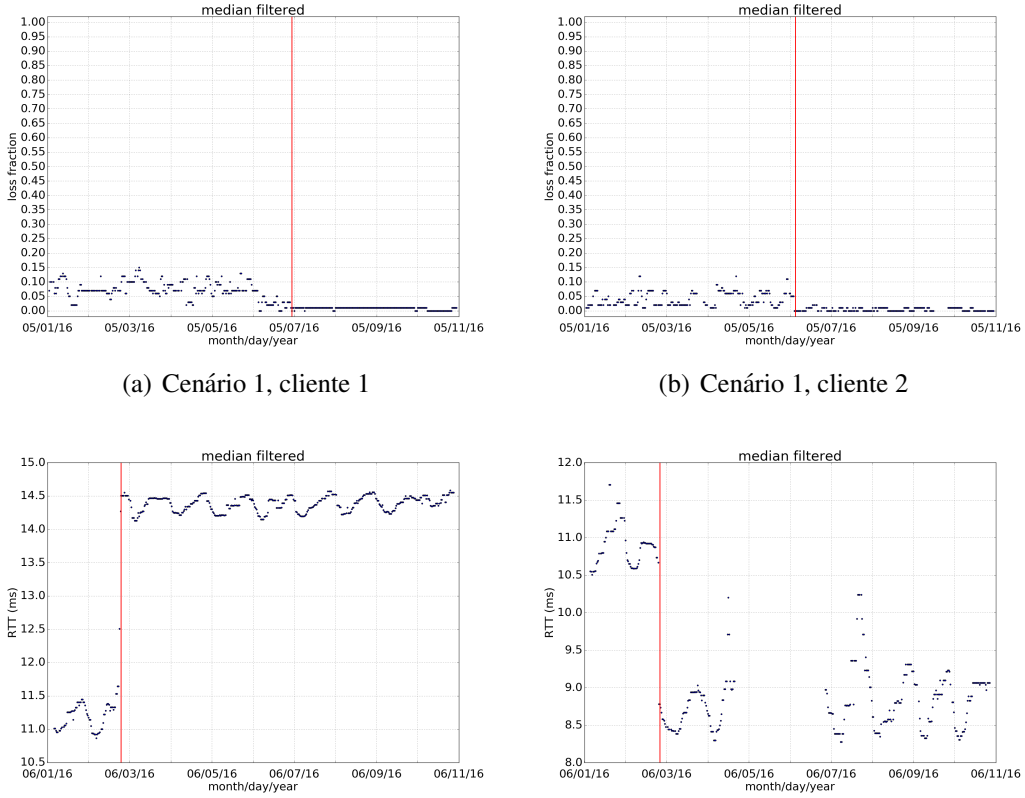
(a) Cliente 1 (b) Grupo de usuários identificado

Figura 8. Evento identificado em um único vértice

A Figura 8(a) exemplifica um evento detetado em um único vértice com grau de entrada igual a zero. Neste caso nenhuma correlação espacial com outros vértices foi identificada, o que indica que o problema é específico da região determinada pelo vértice cinza na Figura 8(b).

As Figuras 9(a) e 9(b) mostram a série temporal de perda de pacotes de 2 clientes de um mesmo vértice de grau de entrada zero. Visualmente ambas indicam um *change point* aparentemente no mesmo instante. No entanto, o cliente representado pela figura da esquerda detetou o *change point* no fim do dia 07/05/2016, enquanto o outro cliente, representado pela figura da direita, apresentou a mudança no início do dia. A diferença de tempo entre as mudanças foi maior do que a capturada pelo parâmetro δ, ou seja, o sistema identificou cada mudança como um evento distinto. Não podemos afirmar então se trata-se de um único evento ou de 2 eventos distintos. Este exemplo mostra a dificuldade de ajustar os parâmetros do algoritmo sem a existência de um *ground truth*.

Um comportamento contra-intuitivo é exemplificado nas Figuras 9(c) e 9(d), que apresentam duas séries temporais de latência de clientes de um mesmo vértice com grau de entrada zero. Ambos os clientes apresentaram uma mudança de comportamento em um período de tempo muito próximo, mas os resultados de um cliente apresentaram melhora enquanto o outro cliente apresentou piora nos seus resultados. Neste mesmo período foi identificado um comportamento similar em outros clientes que não compartilham os mesmos atributos (servidores diferentes, caminhos de rede distintos, estados da federação



(a) Cenário 1, cliente 1

(b) Cenário 1, cliente 2

(c) Cenário 2, cliente 1

(d) Cenário 2, cliente 2

Figura 9. Cenário 1: Clientes com mesmo comportamento e intervalo fora da janela de correlação temporal. Cenário 2: Clientes de mesmo grupo com comportamentos distintos durante mesmo evento.

diferentes).

O comportamento ilustrado nas Figuras 9(c) e 9(d) pode indicar a presença de um evento global na rede que afeta diferentes regiões da rede, como por exemplo uma mudança centralizada nos parâmetros do protocolo de acesso. Como não tivemos acesso a nenhum *dataset* de eventos não foi possível confirmar a causa deste comportamento.

Vários eventos relacionados à vazão *upstream* corresponderam a deteção de um aumento substancial no valor da vazão coletada. A Figura 10 mostra um destes casos. Este tipo de evento indica mudanças na largura de banda contratada pelo assinante.

Por fim, a Figura 11 mostra um dos problemas encontrados durante a execução do *framework* proposto na rede real. É possível perceber que a presença de clientes nesta região da rede é esparsa, o que faz com que diversos agrupamentos possuam apenas um único cliente. Neste caso não foi possível identificar de maneira precisa em qual vértice o evento de rede estava localizado. Por outro lado, é possível identificar a região do possível problema que é o conjunto de nós cinzas na Figura 11(b). A escolha bem planejada dos pontos em que a rede é monitorada evita a ocorrência deste tipo de problema.

A Tabela 1 sumariza os eventos identificados pelo *framework* de análise de dados. Ela classifica os eventos conforme o tipo, melhoria ou piora, de acordo com o item 2 do

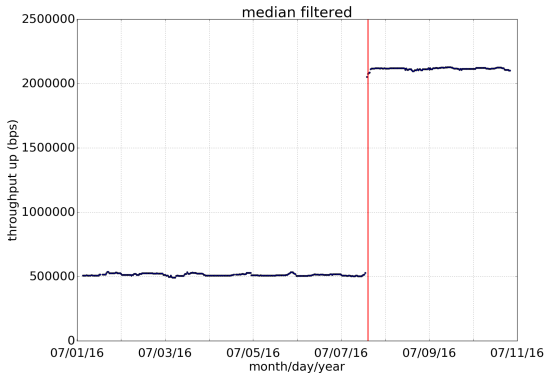


Figura 10. Possível mudança na banda contratada.

início da seção 3. Note-se que nos estudos que fizemos sobre os mais de dois mil clientes foram detetados milhares de eventos em cada uma das séries de latência, taxa de perda de pacotes e vazão máxima *upstream*.

Métrica	Eventos	
	Melhoria	Piora
Latência bidirecional	1520	1532
Taxa de perda bidirecional	310	262
Vazão máxima *upstream*	730	641

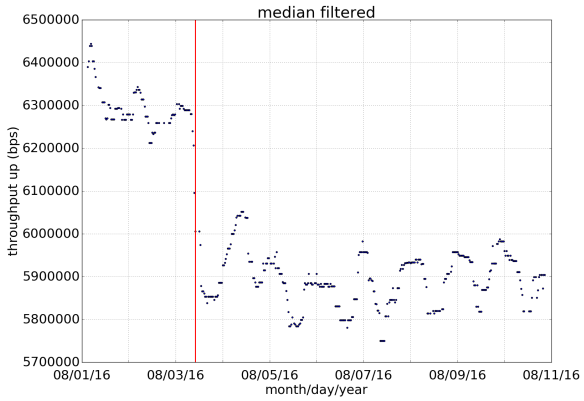
Tabela 1. Resumo dos eventos detetados

5. Trabalhos Relacionados

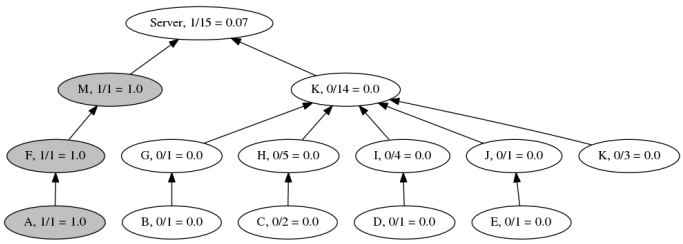
Outros trabalhos na literatura tratam do problema de deteção e localização de falhas em redes de computadores. Em [Yan et al. 2012] é definido o sistema Argus, que utiliza a informação global da rede e dados coletados passivamente para inferir a qualidade de serviço fim-a-fim. A identificação e localização de problemas é realizada a partir de técnicas de agregação temporal e agregação espacial. Diferentemente do trabalho proposto, o sistema Argus utiliza técnicas de deteção de anomalia para a identificação de eventos e não possui uma capilaridade tão grande quanto a obtida pelas medições deste trabalho.

O projeto NetNorad [Adams et al. 2016] foi desenvolvido pelo Facebook com o objetivo de automatizar a análise de falhas de rede na infraestrutura da empresa. A ferramenta utiliza medições de latência para diferentes pontos da rede utilizando ping UDP e aplica técnicas de agregação espacial para a localização de problemas. Esta ferramenta é aplicada em um contexto diferente do utilizado neste trabalho, que coleta as informações a partir de gateways residenciais e portanto abrange uma capilaridade bem maior do que uma rede interna da empresa.

Em [Choffnes et al. 2010] é proposto o *framework* de monitoramento de rede CEM. A ferramenta funciona a partir de medições realizadas por um *software* que é executado em *hosts* de usuários finais. Cada *host* captura métricas de desempenho relacionadas a um serviço específico, como vídeo sob demanda, sendo também responsabilidade dos *hosts* a identificação dos problemas locais. Após a análise local, os *hosts* enviam as informações obtidas para um sistema de armazenamento distribuído. A partir das



(a) Cliente 1



(b) Grupo de usuários identificado

Figura 11. Dificuldade de localização do evento para clientes esparsos.

informações de diversos clientes as falhas de rede são localizadas utilizando um modelo estatístico, correlacionando os resultados de maneira espacial e temporal. Medições realizadas por aplicações em *hosts* possuem diversos fatores que diminuem a robustez dos resultados, como interferência de outras aplicações concorrentes, tráfego concorrente e problemas no sinal de redes sem fio. Além disso, o *framework* CEM possui foco maior em medições passivas realizadas quando a aplicação está em uso, de forma que a frequência das medições depende do comportamento de cada usuário.

O nosso trabalho se difere da literatura quando consideramos que as métricas de qualidade são obtidas a partir de *gateways* residenciais presentes na borda da rede, que geram medições contínuas, robustas e com uma capilaridade muito maior do que a considerada nos outros trabalhos. A utilização de técnicas de *change point detection* também difere este trabalho de sistemas como o Argus, que utilizam deteção de anomalia para a identificação dos eventos.

6. Conclusão

Este trabalho propõe um *framework* de análise de dados para deteção e localização de eventos de rede. Com este objetivo, são detetadas mudanças estatísticas em séries temporais de métricas de QoS coletadas de diferentes clientes, as quais são correlacionadas utilizando a topologia inferida a partir das medições de *traceroute*. Os resultados mostram que é possível identificar e localizar os eventos utilizando apenas as métricas de QoS fim-a-fim e as medições de *traceroute* de diferentes clientes. Apesar deste ser um primeiro estudo, ele foi feito com séries reais de milhares de clientes voluntários. Em trabalhos futuros, pretendemos fazer uma comparação detalhada de diferentes métodos de deteção

de *change point*.

Métricas de qualidade de serviço fim-a-fim tem como principal vantagem a relação direta com a experiência percebida pelo usuário. Uma possível extensão deste trabalho é a criação de métodos para a priorização dos eventos de rede que causem maior impacto na qualidade de experiência do usuário final. Também é de interesse a análise da correlação entre as diferentes métricas de QoS utilizadas para a deteção de eventos.

Por fim, a criação de um *dataset* de eventos de rede tornaria possível a realização de uma análise supervisionada para a escolha dos parâmetros do modelo. O presente trabalho mostrou que o uso de técnicas de *change point detection* é uma ferramenta viável para detetar problemas em redes de ISPs.

Referências

Adams, A., Lapukhov, P., and Zeng, J. H. (2016). NetNORAD: Troubleshooting networks via end-to-end probing. `https://code.facebook.com/posts/1534350660228025/netnorad-troubleshooting-networks-via-end-to-end-probing/`.

Adams, R. P. and MacKay, D. J. (2007). Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*.

Aminikhanghahi, S. and Cook, D. J. (2017). A survey of methods for time series change point detection. *Knowledge and information systems*, 51(2):339–367.

Choffnes, D. R., Bustamante, F. E., and Ge, Z. (2010). Crowdsourcing service-level network event monitoring. *SIGCOMM Comput. Commun. Rev.*, 40(4):387–398.

Fearnhead, P. (2006). Exact and efficient bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16(2):203–213.

Kehagias, A. (2004). A hidden markov model segmentation procedure for hydrological and environmental time series. *Stochastic Environmental Research and Risk Assessment*, 18(2):117–130.

Kifer, D., Ben-David, S., and Gehrke, J. (2004). Detecting change in data streams. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB '04, pages 180–191. VLDB Endowment.

Luong, T. M., Rozenholc, Y., and Nuel, G. (2013). Fast estimation of posterior probabilities in change-point analysis through a constrained hidden markov model. *Computational Statistics and Data Analysis*, 68:129 – 140.

Maidstone, R., Hocking, T., Rigaill, G., and Fearnhead, P. (2017). On optimal multiple changepoint algorithms for large data. *Statistics and Computing*, 27(2):519–533.

Montañez, G. D., Amizadeh, S., and Laptev, N. (2015). Inertial hidden markov models: Modeling change in multivariate time series. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 1819–1825. AAAI Press.

Parhami, B. (1994). Voting algorithms. *IEEE transactions on reliability*, 43(4):617–629.

Su, W.-x., Zhu, Y.-l., Liu, F., and Hu, K.-y. (2013). On-line outlier and change point detection for time series. *Journal of Central South University*, 20(1):114–122.

Truong, C., Oudre, L., and Vayatis, N. (2018). A review of change point detection methods. *arXiv preprint arXiv:1801.00718*.

Yan, H., Flavel, A., Ge, Z., Gerber, A., Massey, D., Papadopoulos, C., Shah, H., and Yates, J. (2012). Argus: End-to-end service anomaly detection and localization from an isp's point of view. In *2012 Proceedings IEEE INFOCOM*, pages 2756–2760.