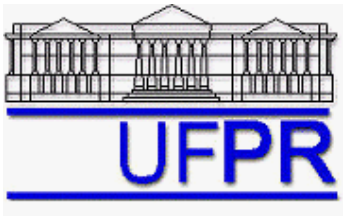

UNIVERSIDADE FEDERAL DO PARANÁ
SETOR DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
MESTRADO EM TELECOMUNICAÇÕES

Disciplina: COMUNICAÇÃO DE DADOS

Professor: Eduardo Parente Ribeiro

AMOSTRAGEM DE TRAJETÓRIA PARA OBSERVAÇÃO DIRETA DE TRÁFEGO

Vilson Rodrigo Mognon

Curitiba
Outubro de 2001

AMOSTRAGEM DE TRAJETÓRIA PARA OBSERVAÇÃO DIRETA DE TRÁFEGO

RESUMO: *Este trabalho compreende um resumo e complemento do artigo "Trajectory Sampling for Direct Traffic Observation" desenvolvido por N. G. Duffield e publicado na IEEE Transaction on Networking em junho de 2001* [ND]. *Tal artigo trata de um processo de medida tráfego, um componente crítico para o controle e engenharia de redes de comunicação. No artigo foi discutido que a medida de tráfego deveria tornar possível a obtenção do fluxo espacial de tráfego pelo domínio, isto é, os caminhos seguidos pelos pacotes entre qualquer ponto de ingresso e de egresso do domínio* [ND]. *A maioria da alocação de recurso e capacidade de planejamento de tarefas podem beneficiar de tal informação. Também, deveriam ser obtidas medidas de tráfego sem um modelo de roteamento e sem conhecimento do estado de rede. Isto permite o processo de medida de tráfego ser ajustável as falhas da rede e incerteza de estado. Foi proposto um método que permite a inferencia direta de fluxos de tráfego através de um domínio observando as trajetórias de um subconjunto de todos os pacotes que atravessam a rede* [ND].
As vantagens chaves do método são que 1) não conta com o estado de roteamento; 2) seu custo de implementação é pequeno; e 3) a medida que informa tráfego é modesta e pode ser controlada precisamente. A idéia chave do método é testar pacotes baseado em uma "hash function" computada em cima do conteúdo do pacote. Usando a mesma "hash function" levará a mesma amostra de um conjunto de pacotes no domínio inteiro, e nos permite reconstruir as trajetórias do pacote.

1 INTRODUÇÃO

A eficiência da alocação de recurso e a qualidade de serviço (QoS) contanto através de redes IP depende criticamente da efetiva administração de tráfego. Administração de tráfego consiste de controle de tráfego a curto prazo e engenharia de tráfego a longo-prazo. Controle de tráfego opera em uma escala de tempo de segundos e sem intervenção humana direta. Exemplos de funções de controle de tráfego incluem controle de congestionamento, recuperação automática no caso de falhas de links ou roteador, ou controle de admissão. Engenharia de tráfego opera na escala de minutos, horas, semanas ou meses, e tipicamente com algum grau de intervenção humana. Sua meta é otimizadamente alocar recursos de rede, como capacidade de link, para classes diferentes de tráfego de rede para assegurar boa qualidade de serviço e alta eficiência de rede. Exemplos de funções de engenharia de tráfego incluem a caracterização de tráfego, contabilidade, e planejamento de capacidade e aprovisionamento.
Todas estas funções representam laços de realimentação em grande alcance na escala de tempo e na variação espacial de extensão, e da observação ou medida de tráfego. É portanto um componente integral destas funções. A importância de capacidades de medida de tráfego é composta pelo fato que redes IP não mantêm estado per-fluxo. Por contraste, em redes comutadas por circuito, o tráfego é essencialmente "observable for free" porque estados por-chamada existe ao longo de cada nó no caminho da chamada. De certo modo, a escalabilidade das redes IP sem estado tem sido comprada ao custo da observabilidade.
Virtualmente todas as funções de engenharia de tráfego, como otimização de rota, ou planejando de estratégias de sobrefalhas, contam com uma compreensão do fluxo espacial de tráfego pelo domínio. Por exemplo, suponha que observamos que algum link na backbone está sobrecarregado. A ação corretiva apropriada requer o entendimento de qual

ponto este tráfego observado neste link é originado e onde é encabeçado, que clientes são afetados pela congestionamento; sem esta informação, soluções efetivas (por exemplo, reencaminhando de parte daquele tráfego) não pode ser tomadas [11], [12]. Também, deveria ser possível deduzir que fração de tráfego que entra no domínio de medida a um certo ponto de ingresso atravessando cada link na rede, por exemplo, enfocar em como o tráfego de cliente específicos fluem pelo domínio, e diagnosticar qual link poderia ser a rasão para um problema de desempenho experimentado por aquele cliente.

Informação de tráfego espacial do grande domínio também é uma condição prévia para o estabelecimento de túneis para comutação de etiqueta [3], ou decidir qual ponto de ingresso potencial é melhor para conectar um cliente novo no domínio.

Foi distinguido entre métodos direto e indireto de medida. Conceitualmente, um método de medida indireta conta com um modelo de rede e informação de estado de rede para deduzir o fluxo espacial de tráfego pelo domínio. Por exemplo, suponha que o tráfego só é observado nos pontos de ingresso da rede(por exemplo, computando estatísticas na distribuição de pares fonte-destino) [ND]. Para deduzir como aquele tráfego flue pelo domínio, informação oportuna e precisa sobre o estado do protocolo de roteamento e estados dos links tem de estar disponíveis. Se suposições sobre roteamento de tráfego têm que ser feitas para obter a matriz de fluxo de tráfego, então o uso de uma tabela de roteamento antiquada pode conduzir a conclusões errôneas, e portanto alocação não ótima de recursos de rede.

Geralmente, métodos de medida indiretas sofrem a incerteza associada com o estado físico e lógico de uma grande rede heterogênea [11]. Esta incerteza tem várias fontes. Primeiro, o comportamento exato de um elemento de rede, como um roteador, não é conhecido exatamente ao provedor de serviço e depende em escolhas de específicas de projeto. Por exemplo, o algoritmo para divisão de tráfego entre vários caminhos abertos menores (OSPF) não é padronizado. Segundo, há fontes deliberadas de aleatoriedade na rede para evitar acidental sincronização, por exemplo, cronômetros randomicos em protocolos de roteamento[14].

Terceiro, alguns comportamentos da rede depende de eventos fora do controle do domínio, por exemplo, como o tráfego é roteado dentro de um sistema autônomo (AS) depende em parte da dinâmica do anúncio da rota para este AS por domínios vizinhos [18].

Quarto, a interação entre esquemas adaptáveis, operando a diferentes escalas de tempo e níveis de localidade (por exemplo, QoS de roteamento, controle de congestionamento fim-a-fim) simplesmente podem ser muito complexo de caracterizar e predizer [30]. Finalmente, com o aumento do tamanho e complexidade, aumenta-se probabilidade de falhas e mal configuração romperem a operação normal da rede. Freqüentemente, a medida de tráfego é um das ferramentas potenciais para descobrir e diagnosticar tais problemas; porém, este benefício é mitigado se medida de tráfego requer correta operação de rede.

Um método direto não conta com um modelo de rede e uma estimação de seu estado e seu comportamento esperado. Porém, confia na observação direta de tráfego em pontos múltiplos na rede. Como tal, não sofre das fontes de incerteza discutidas acima. Neste trabalho, foi descrito um método direto para medida tráfego, chamado amostragem de trajetória [ND]. O método amostra pacotes que atravessam cada link (ou um subconjunto destes links) dentro um domínio de medida. O subconjunto de pacotes amostrados em cima de um certo período de tempo pode ser usado então como um representante do tráfego global.

Se simplesmente fossem amostrados aleatoriamente pacotes a cada link, então, estaríamos impossibilitados de derivar o caminho preciso que um pacote amostrado seguiu pelo domínio desde o ponto de ingresso até o egresso. A idéia chave é então basear a decisão

amostragem em uma "hash function" deterministica em cima do conteúdo de pacote. Se a mesma "hash function" é usada ao longo do domínio para amostrar pacotes, então estamos assegurados que um pacote ou é amostrado em todo link que atravessa, ou em nenhum link. Em outras palavras, pode-se efetivamente coletar amostras de trajetória de um subconjunto de pacotes. A escolha de uma "hash function" apropriada será obviamente crucial para assegurar que este subconjunto não é estatisticamente influenciado de qualquer forma. Para isto, o processo de amostragem, embora uma função deterministica do conteúdo do pacote, tem que assemelhar a um processo randomico de amostragem.
Um segundo ingrediente da proposta é o etiquetamento do pacote. Note que para obter amostras de trajetórias, não estamos interessados no conteúdo de pacote; simplesmente precisamos saber que algum pacote atravessou um conjunto de links. Mas para saber isto, é suficiente obter um identificador único de pacote, ou etiqueta, para cada pacote amostrado dentro do domínio e dentro de um período medida. Como a etiqueta é única, saberemos que um pacote atravessou o conjunto de links que reportaram aquela etiqueta particular. Usa-se uma segunda "hash function" para computar etiquetas de pacote que são, com probabilidade alta, únicas dentro um período de medida. Enquanto o tamanho da etiqueta do pacote obviamente depende da situação específica, note que a etiqueta pode ser em prática bastante pequena (por exemplo, 20 bit). Como o tráfego de medida que tem que ser coletado do domínio consiste somente de tais etiquetas (mais um pouco de informação auxiliar), o overhead para coletar amostras de trajetória é pequeno. Amostragem de trajetória tem várias vantagens importantes. É um método direto para medida de tráfego, e como tal não requer qualquer informação de estado de rede. O fluxo espacial de tráfego pelo domínio pode ser deduzido de amostras de trajetória, isto é, caminhos tomados por um subconjunto de pacotes pseudorandomicos pelo domínio. Amostragem de trajetória não requer estado do roteador senão um pequeno buffer para etiqueta. A quantia de tráfego de medida necessário é modesto e pode ser controlado precisamente. Pacotes de multicast não requerem nenhum tratamento especial - trajetória associada com um pacote de multicast é simplesmente uma árvore em vez de um caminho. Finalmente, amostragem de trajetória pode ser implementado usando processadores digitais de sinal (DSPs) até mesmo para a interface de mais alta velocidade disponível hoje.
Este trabalho é estruturado da seguinte forma: foi definido a notação e a amostragem de trajetória. Discuti-se a escolha de parâmetros para as "hash function", e demonstra-se suas propriedades estatísticas. Da-se um exemplo de medida de tráfego baseado em uma extensa rota de pacotes e finalmente discuti-se a implementação e possível extensões para amostragem de trajetória.

2 DESCRIÇÃO FORMAL DA AMOSTRAGEM DE TRAJETÓRIA

Para simplicidade, foi descrito o esquema que assume que todos os pacotes tem o mesmo tamanho S bits. Foi representado o domínio de medida como um gráfico dirigido G(V,E), onde V é o conjunto de nós e E é o conjunto de links dirigidos. Pacotes entram no domínio de medida em um nó de ingresso. Eles atravessam vários links para deixar o domínio de medida em nó de egresso (ou vários nós de egresso no caso de um multicast packet[1]). Um pacote pode ser potencialmente roteado a um intermédio nó. Deixa-se xi(Pk) denotar o conteúdo de um pacote k a link i, isto é, a sucessão de bits que constituem o cabeçalho IP e o conteúdo do pacote IP. Quando não há nenhum risco de ambigüidade, por exemplo, quando considerando um fluxo de pacotes a um único link, refere-se a um pacote e seu conteúdo intercambiável. Considere todos os pacotes P1,...PN que entram no domínio dentro de um intervalo de medida de duração T. A trajetória do pacote é o conjunto de

links atravessado pelo pacote Pk. No caso de um pacote de unicast, a trajetória é um caminho do nó de ingresso para o nó de egresso ou para o nó onde o pacote é roteado. No caso de um pacote de multicast, a trajetória forma uma árvore derivada do nó de ingresso.

A função de invariança ϕ é uma função do conteúdo de pacote cuja saída depende da invariância do conteúdo de pacote, isto é, os bits do pacote que não são modificado ao remeter, como descrito abaixo.

Uma função de invariança não depende, por exemplo, no tempo de vida (TTL) campo que é decrementado a cada salto. Sem perda de generalidade, assume-se aqui que a função ϕ extrai simplesmente todos os Sc bits invariantes do pacote.

$$\phi : \{0,1\}^{S} \rightarrow \{0,1\}^{Sc} \quad (1)$$

A idéia básica da amostragem da trajetória é decidir se para amostrar um pacote P baseado em uma função deterministica do conteúdo invariante do pacote ϕ *(x(P))*; chama-se esta função deterministica de "hash function" de amostragem , definida como um mapa,

$$h: \{0,1\}^{Sc} \rightarrow \{0,1\}^{l} \quad (2)$$

do conteúdo invariante do pacote dentro dos *l*-bits números binários. Um pacote P é amostrado se $h(\phi(x))$ pertence *D* por algum dado dominio de amostragem $D \subset \{0,1\}^{l}$. Chama-se a função indicadora h_D definida por:

$$h_D(\phi(x)) = \begin{cases} 1, & \text{se } h(\phi(x)) \in D \\ 0, & \text{caso contrário} \end{cases} \quad (3)$$

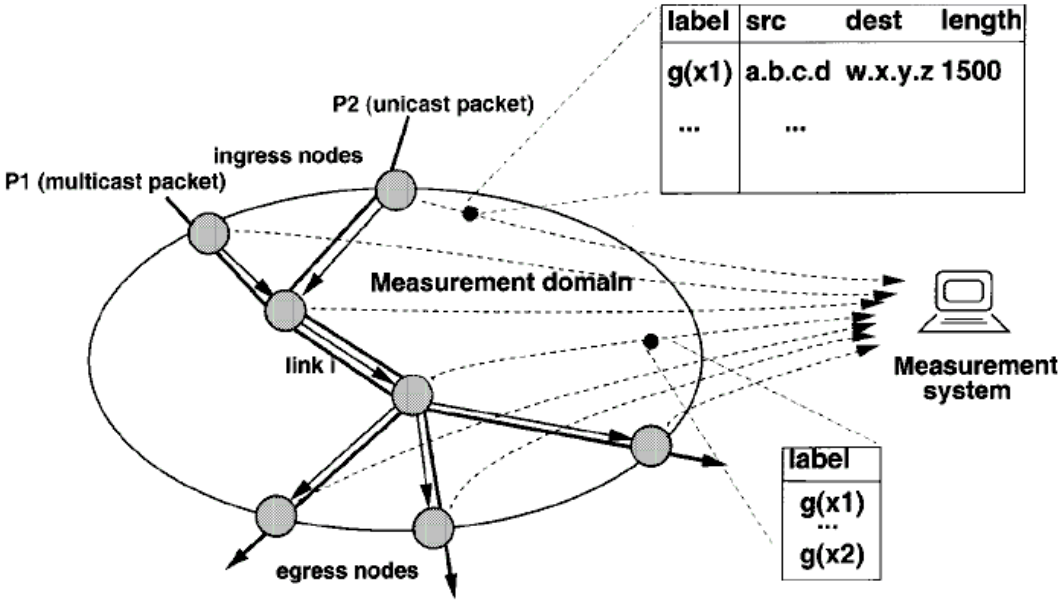

Figura 1. Representação esquemática da amostragem de trajetória. Um sistema de medida coleta etiquetas de pacote de todos os links dentro do domínio. Etiquetas só são coletadas de um pseudorandomico subconjunto de de todos os pacotes que atravessam o domínio. Ambos a decisão se amostrar um pacote ou não, e a etiqueta do pacote, é uma função do conteúdo invariante do pacote [ND].

Note que foi usado a mesma função de amostragem em cada link no domínio de medida[ND]. Deste modo, um pacote ou é amostrado em todos lugares em sua trajetória ou

não é em nenhuma parte, e o dados de amostragem nos deixa reconstruir as trajetórias do pacotes amostrados.
Em princípio, um nó poderia enviar o conteúdo inteiro de um pacote amostrado para o sistema de coleção de medida. Porém, isto é muito ineficiente; note que para identificar trajetórias, não estamos interessados no conteúdo do pacote por si; só precisamos de um identificadror para distinguir um determinado pacote de outros pacotes amostrados, para obter amostras de não ambiguas da trajetórias do pacote.
Então, foi usado uma "hash function" de identificação para computar um identificador de pacote compacto na parte constante do pacote [ND].

$$g: \{0,1\}^{Sc} \rightarrow \{0,1\}^{m} \quad (4)$$

Deste modo, é necessário somente enviar m bits por pacote amostrado por link à estação de coleção.
Uma alternativa de comprimir o cabeçalho do pacote para uso como uma etiqueta provavelmente seja não efetivo ao reduzir volume de etiqueta.
Compressão efetiva está baseado em construir um dicionário de símbolos repetidos nos objetos a ser comprimido. Tal repetição não é esperado que aconteça em pacotes únicos. Uso de um novo dicionário para pacotes múltiplos com campos comuns (por exemplo, pacotes amostrados de um fluxo) requereriam manutenção de estado adicional no roteador. Além disso, o tamanho de etiqueta pode não ser fácil controlar. Através de comparação, uso de uma "hash" etiqueta é simples, sem estado, e fornece etiquetas de tamanho fixo.
Em sua forma mais básica, amostragem de trajetória executa o seguinte operação simples a cada link no domínio: para cada pacote observado de conteúdo x, se $h_D(\phi(x)) = 1$ então envia a etiqueta $g(\phi(x))$ para o sistema de coleção de medidas. Enquanto isto basta identificar trajetórias de pacotes, informação adicional sobre um pacote amostrado (como seu tamanho e seus endereços de fonte e destino) são requerido para muitos propósitos de medida.
É suficiente coletar esta informação adicional uma vez por pacote amostrado. Por exemplo, nós de ingresso poderiam ser configurados para recobrar esta informação com as etiquetas, enquanto todos os outros nós só coletam etiquetas (conforme Figura 1).

2.1 O cabeçalho do datagrama IP

O cabeçalho tem uma parte fixa de 20 bytes e uma parte opcional de tamanho variável [AT]. O formato do cabeçalho é mostrado na figura. O campo VERSION controla a versão do protocolo a que o datagrama pertece. Incluindo-se a versão em cada datagrama, é possível verificar a transição entre as versões, que pode levar meses ou anos com duas máquinas na rede operando com versões distintas.
Como o tamanho do cabeçalho não é constante, exite o campo IHL, que informa o tamanho do cabeçlho em palavras de 32 bits. O valor mínimo é 5, que represena a ausência de opções. O valor máximo desse camop de 4 bits é 15, o que limita o cabeçalho a 60 bytes e o campo de opções a 40 bytes.
O campo TYPE OF SERVICE permite que o host informe à sub-rede o tipo de rede que deseja. São possíveis várias combinações de confiabilidade e velocidade. Em se tratando de voz digitalizada, a entrega rápida vence a entrega segura.
O campo propriamente dito contém um campo PRECEDENTE de três bits, três flags (D, T e R) e 2 bits que não são utilizados. O campo PRECEDENTE tem uma prioridade de 0 (normal) a 7 (pacote de controle de rede). Os bits de três flags permitem que o host especifique o que é mais importante no conjunto { Retardo, Taxa de Transferência, Confiabilidade }. Teoricamente, esses campos permitem que os roteadores optem, por

exemplo, entre uma ligação de satélite com alta taxa de transferência mas com grandes retardos ou uma linha privativa com uma taxa de transferência baixa e retardo pequeno. Na prática, os roteadores ignoram completamente o campo.
O campo TOTAL LENGHT inclui tudo o que há no datagrama - cabeçalho e dados. O tamanho máximo é de 65.535 bytes. O campo IDENTIFICATION é necessário para permitir que o host de destino determine a qual datagrama pertence um fragmento recém-chegado. Todos os fragmentos de um datagrama contêm o mesmo valor de IDENTIFICATION.
Em seguida há um bit não utilizado e dois campos de 1 bit. DF significa Don't Fragment (não fragmente). Trata-se de uma ordem para os roteadores não fragmentarem o datagrama porque a máquina de destino é incapaz de juntar os pedaços novamente. MF significa More Fragments (Mais Fragmentos). Todos os fragmentos, exceto o último, têm esse conjunto de bits, que é necessário para que se saiba quando todos os fragmentos de um datagrama chegaram.
O campo FRAGMENT OFFSET informa a que ponto do datagrama atual o fragmento pertence. Todos os fragmentos de um datagrama, com exceção de último, devem ser múltiplos de 8 bytes, que é a unidade de fragmento elementar. Como são fornecidos 13 bits, existe um máximo de 8,192 fragmentos por datagrama, resultando em um tamnaho máximo de datagrama de 63.636 bytes, um a mais do que o campo TOTAL LENGTH.
O campo TIME TO LIVE é um contador usado para limitar a vida útil do pacote. A princípio representaria o tempo em segundos, mas na prática ele simplesmente conta os hops. Quando o contador chega a zero, o pacote é descartado em um pacote de advertência é enviado para o host de origem. Esse recurso evita que os datagramas fiquem vagando indefinidadmente, algo que aconteceria se as tabelas de roteamento fossem danificadas.
Quando tiver montado um datagram completo, acamada de rede precisa saber o que fazer com ele. O campo PROTOCOL informa a ele o processo de transporte que deverá ser aplicado ao datagrama. O TCP é uma possibilidade, mas também há o UDP e alguns outros. A numeração dos protocolos se aplica a toda a Internet e é definidad na RFC 1700.
O campo HEADER CHECKSUM confere apenas o cabeçalho. Essa soma de verificação é útil para a detecção de erros gerados por palavras de memória danificadas em um roteador. O algoritmo tem como objetivo somar todas as meias palavras de 16 bits à medida que elas chegam, utilizando a aritmética de completmento de 1 e depois calculando o complemento de 1 do resultado. Devido a esse algoritmo, presume-se que HEADER CHECKSUM seja zero no momento de chegada. O uso do algoritmo é mais eficaz do que o acréscimo normal. Observe que HEADER CHECKSUM deve ser recontado a cada hop, pois pelo menos, um campo sempre se altera (TIME TO LIVE).
Os campos SOURCE ADRESS e DESTINATION ADDRESS indicam o número da rede e o número do host, ou seja o endereçpo IP de origem e destino do datagrama.
O campo OPTIONS foi projetado para permitir que versões posteriores do protocolo incluam informações inexistentes no projeto original, possibilitando a experimentação de novas idéias e evitando a alocação de bits de cabeçalho para informações raramente necessárias. Existem opções de tamanhos variáveis. Cada uma começa com um código de um byte identificando a opção. Atualmente, há cinco opções definidas: Security, Strict Source Routing, Loose Source Routing, Record Route, Timestamp, mas nem todos os roteadores acetiam todas essas opções.

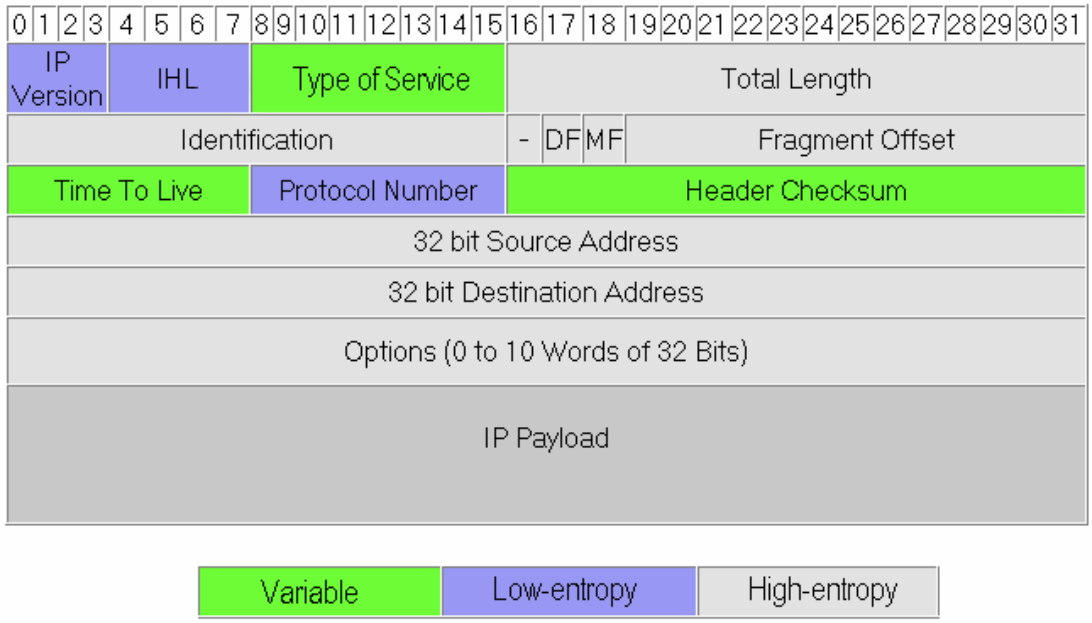

Figura. 2.Pacote de conteúdo invariante. As "hash functions" são computadas em cima de um subconjunto de campos de cabeçalho e parte do payload. Campo variáveis mudam ao longo do caminho; campos de baixo-entropia são invariantes ao longo de caminhos mas variam pouco entre pacotes; campos de alta entropia são invariantes ao longo de caminhos e variam significativamente entre pacotes [WW].

2.2 Identidade de Pacote e Conteúdo Invariante

A definição da função invariancia ϕ é completada pela identificação do conteúdo invariante do pacote. Foram considerados apenas pacotes IP versão 4. Na Figura 2, ilustra-se os campos do cabeçalho do pacote IP e o pacote payload. Divide-se os campos em três categorias: 1) campos variáveis, isto é, aqueles que mudança ao longo do caminho; 2) campos de baixo-entropia, isto é, aqueles que são invariante para um determinado pacote ao longo de um caminho, mas tem pequena ou nenhuma variação de pacote-a-pacote; e 3) campos de alto-entropia, que também são invariantes para um determinado pacote ao longo de um caminho e podem variar grandemente entre pacotes.

O cabeçalho invariante do pacote podem então ser tomado como o conjunto de campos de alta entropia. Os campos de baixo-entropia também poderiam ser incluídos, mas isto adicionaria escassa variabilidade estatística para o conteúdo invariante do pacote, e requereria tempo de processamento adicional quando calculando a amostragem e etiquetamento.

Os campos variáveis são o TTL (bits 64-71) o qual é decrementado por salto, e o campo SERVICE TYPE (bits 8-15) pois alguns de seus bits podem ser mudados em trânsito, por exemplo, durante Notificação de congestionamento Explícito [21], e através de operação de Serviços Diferenciados [4]. O HEADER CHECKSUM (bits 80-95) é recalculado em mudanças de cada um destes e conseqüentemente também é variável.

Campos de baixo-entropia são a VERSION (bits 0-3), HEADER LENGHT (bits 4-7), e PROTOCOL (bits 72-79). Estes são constante ou tomam um de um número pequeno de valores.

Os campos restantes são tidos de alta entropia. SOURCE OR DESTINATION IP ADRESS (bits 96-159) são incluído no conteúdo invariante do pacote. É incluído também o Campo

de IDENTIFICATION (bits 32-47). FLAGS (bits 48-51) e FRAGMENT OFFSET (bits 52-63) são igualmente mutáveis através da fragmentação. Realmente, a fragmentação eleva potencialmente uma maior edição, desde que fornece um mecanismo por qual a noção de um único pacote identificável torna-se corrompido. Porém, esperamos que a fragmentação seja confinada à extremidade da rede, com uma noção de borda-a-borda da identidade de pacote permanecendo válida, até mesmo, em casos onde o fim-a-fim é nulo. Neste caso, podemos incluir TOTAL LENGHT, FLAGS, e FRAGMENT OFSSET dentro do conteúdo invariante.
O resto do pacote que segue os primeiros 20 bytes, completa o conteúdo invariante do pacote. Em certos pacotes de opções IP, como pacotes com uma opção de rota gravada, estes bytes seguintes podem mudar salto por salto. Porém, desde que tais pacotes são raros, acredita-se que o efeito na amostragem pode ser ignorado.

2.3 Trajetórias Ambíguas

Foi discutido como deduzir trajetórias das etiquetas coletadas da rede dentro de um período de medida[ND]. O período de medida é escolhido como um salto superior a vida de pacote. Assume-se que todas as observações do pacote feitas dentro do mesmo período de medida só pode ser distinguido pela etiqueta deles, não pelo tempo de chegada deles dentro do período de medida.
Como etiquetas são alocadas pseudorandomicalmente para pacotes amostrados, há obviamente uma chance de colisão de etiqueta, isto é, de duas ou mais trajetórias de pacotes que têm a mesma etiqueta no mesmo período de medida. A pergunta que se faz nesta subseção é sobre quais circunstâncias pode-se eliminar a ambiguidade destas trajetórias.
É útil introduzir o conceito de um subgráfico de etiqueta associado com uma etiqueta e um período de medida. O subgráfico de etiqueta simplesmente é o gráfico do domínio de rede onde cada link é anotado com o número de etiqueta de tempos i foi gerado por aquele link no período de medida; links com zero são apagado. Um subgráfico de etiqueta basicamente representa a superposição de todas as trajetórias no período de medida que tiveram esta etiqueta.
Restringe-se esta discussão para pacotes de unicast e para subgráficos aciclicos. Primeiro, note que no caso trivial onde um subgráfico de etiqueta origina de de uma única trajetória, esta trajetória pode sempre ser deduzidos não de forma ambigua. Intuitivamente, isto é porque um pacote ou é amostrado em todos lugares no domínio ou em nenhuma parte. Assim, se observamos a etiqueta i exatamente num link inbound e num link outbound de um nó, deve ser o mesmo pacote. Por indução, a trajetória inteira pode ser reconstruída sem ambigüidade.

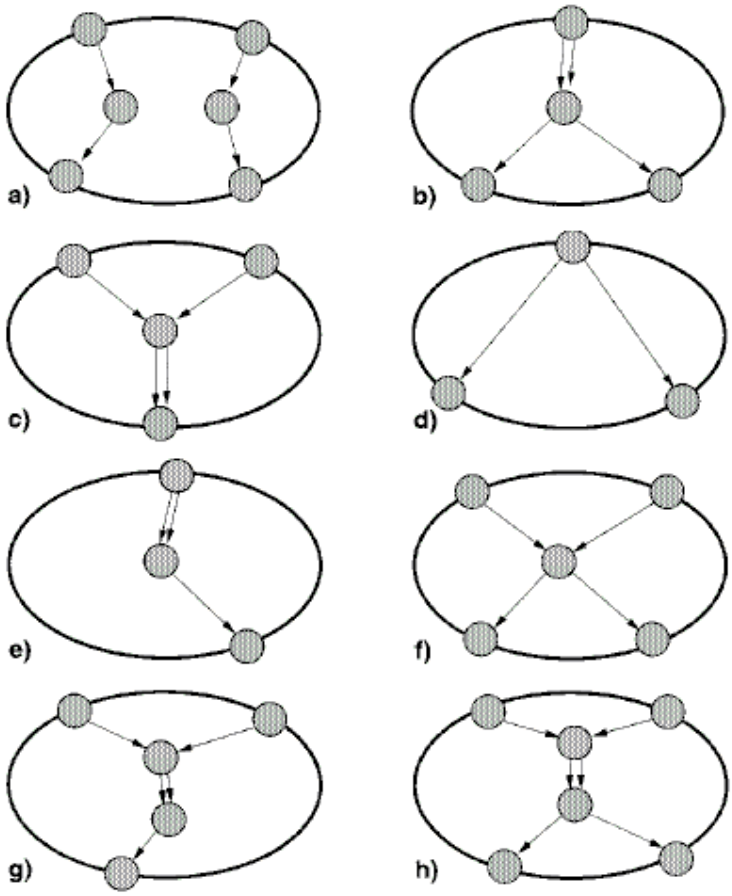

Figura. 3. Eliminação de ambiguidade da trajetória. Exemplos subgráficos de etiqueta não ambiguos (a-e) e ambíguo (f-h). Para (e) e (g), um pacote é roteado a um nó interior[ND].

Segundo, consideraremos o caso onde o subgráfico de etiqueta é a superposição de várias trajetórias. Alguns exemplos de superposição de duas trajetórias são determinadas na Figura 3. Os exemplos (a) até (e) são não ambiguos, enquanto exemplos (f) até (h) são ambíguos.

A seguinte propriedade assegura: um subgráfico de etiqueta é não ambiguo se cada componente conectado ao subgráfico é ou 1) uma árvore de fonte, ou 2) uma árvore sorvedoura tal que para cada nó na árvore sorvedoura, o grau do outbond link é a soma dos graus dos inbounds links. Note que o exemplo (e) não é unambiguous porque o único componente conectado é uma árvore de fonte; também é uma árvore sorvedoura, mas a condição de grau não é satisfeita. Note também que a ambigüidade como definida aqui só pertence para trajetórias seguidas por pacotes. Por exemplo, exemplo (e) é não ambiguo porque não há nenhuma ambigüidade sobre as duas trajetórias seguido pelos pacotes. Porém, se coletarmos outros atributos dos dois pacotes (no nó de ingresso, digamos), então não temos nenhum modo de saber do exemplo (e) qual pacote foi derrubado no meio, e qual pacote foi para o nó de egresso. Em contraste, há vários possíveis conjuntos de trajetórias que podem resultar em ambiguidade, como nos subgráficos de etiqueta (f) até (h). Em resumo, uma etiqueta pode ser atribuída a sua trajetória 1) se a etiqueta é única, ou 2) se pode eliminar a ambiguade. Obviamente, a probabilidade que uma etiqueta de alguma trajetória pode eliminar a ambiguade depende da topologia da rede, e a taxa de tráfego em

todos as outras trajetórias. Então, o número de etiquetas de não ambiguas em uma trajetória é em geral um estimator parcial da taxa de tráfego naquela trajetória, e é necessário a renormalizar estimators de taxa depois de desambiguação. De outra forma para evitar preconceito é tão simplesmente descartar toda as etiquetas duplicadas, independente se elas poderiam ou não ser desambiguadas. Isto é simples, mas implica em algumas perdas de amostras.

2.4 Desempenho da Amostragem de Trajetória

A meta global é obter tantas amostragens de trajetórias pseudorandomicas da rede quanto possível, sem usar, muitos recursos (largura de banda da rede, sistema de coleção, memória). Neste trabalho, foi demonstrado "hashes" baseado em aritmética modular (veja, por exemplo, [17]), e mostrado que os parâmetros deste esquema pode ser escolhido tal que os "hashes" aparecem estatisticamente independente do conteúdo de pacote original, habilitando assim a amostragem imparcial. Computou-se a ótima escolha do número total de amostras a ser coletado da rede e o número de bits por amostra, sujeito a uma limitação de largura de banda da rede disponível para medida de tráfego.

2.5 Especificação de Funções Hash

Foram considerados os bits ordenados de um pacote x e de sua parte invariante $\phi(x)$ como inteiros binários. Foi usado a amostragem hash

$$h(\phi(x)) = \phi(x) \bmod A \qquad (5)$$

e domínio de amostragem D = {0,1,2,...,r-1} para inteiros positivos, A e r. Assim um pacote é amostrado se $\phi(x) \bmod A$ é menor que r. O modulo de A é escolhido para evitar colisões que surgem de certas propriedades estruturais dos conteúdos de pacote. Por exemplo, espera-se achar conjuntos complementares de pacotes no qual endereços IP fonte e destino são intercambiados e surgem do fluxo de dois-pontos no tráfego em sessões TCP. A função hash, e conseqüentemente os modulos, deve ser escolhido de forma a evitar colisões em qual um par de pacotes que pouco diferem entre si, como um intercâmbio de campos que é detectado pelo o mesmo resto de divisão. Knuth (veja [17, sec. 6.4]) formula um condição para vacância de tais colisões, $q^k \pm a \neq 0 \bmod A$ para pequeno a, k onde q é um radix do alfabeto usado para descrever o cabeçalho. Incluindo $q^k = 2^{32}$ neste critério suprime-se colisões do tipo descritas acima. Modulos obedecendo estas condições podem ser selecionados de tabelas de números primos. r determina a granularidade da amostragem; deve ser escolhido suficientemente grande para que a menor taxa de amostragem disponível, isto é, 1/A para r = 1, seja suficientemente pequeno. São codificados pacotes amostrados usando uma "hash function" semelhante

$$g(\phi(x)) = \phi(x) \bmod B \qquad (6)$$

com o modulo de $B \neq A$ para que o hash de identificação seja não correlacionado com a amostragem de pacote.

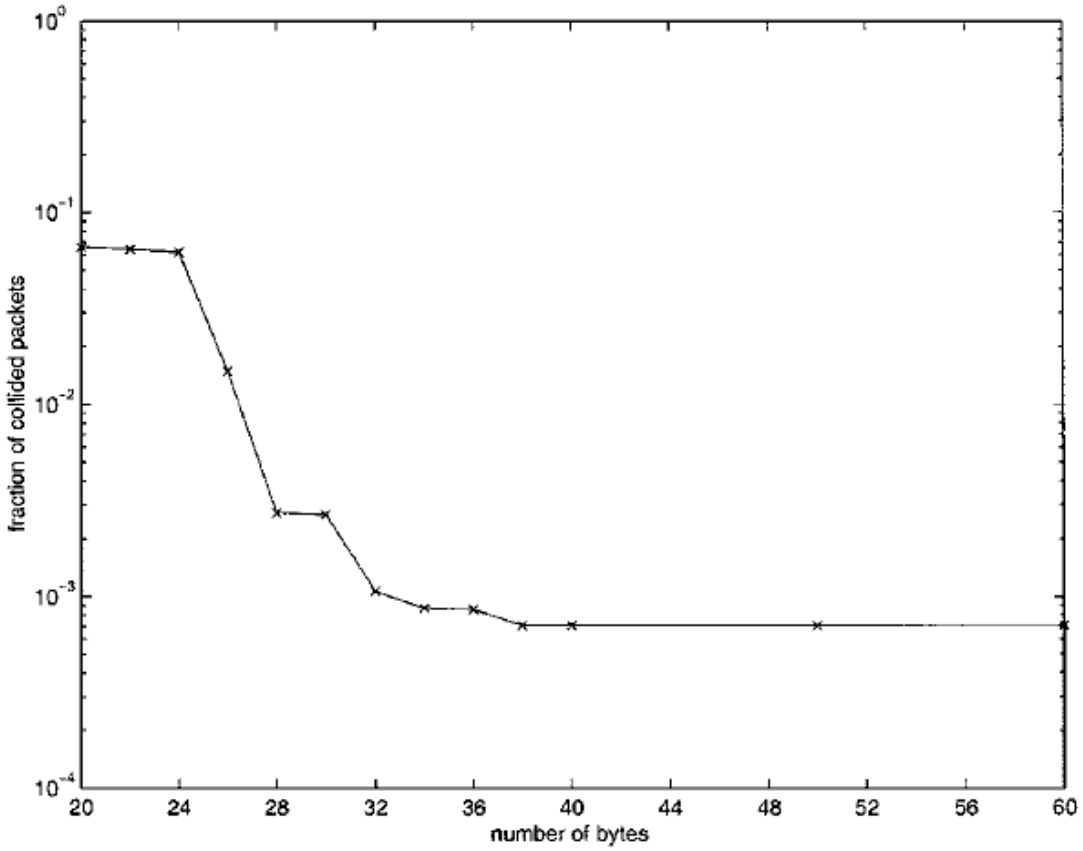

Figura 4. Colisões de pacotes. A fração de pacotes cujo prefixo não único, como uma função de tamanho de prefixo l. O menor valor para o tamanho de prefixo (20 bytes) corresponde ao uso só do cabeçalho do pacote [ND].

2.6 Pacotes Idênticos

Como a função hash é uma função deterministica, se dois pacotes são exatamente idênticos, então a decisão de amostragem e suas etiquetas serão também idênticas. Então, pacotes idênticos não são amostrados pseudorandomicamente pelo método que pode conduzir a estimadores parciais. Temos que se convencer então que pacotes idênticos são raros na prática. Chama-se a ocorrência de pacotes idênticos em colisões de rota.

Mais geralmente, estamos interessados na freqüência com que um prefixo de um certo tamanho *l* (isto é, os primeiros bytes, com o campos variáveis fora) de um pacote não são unicos dentro de um grande conjunto de pacotes. Se podemos identificar um tamanho de prefixo de pacote para qual as colisões são raras, então é suficiente computar a amostragem e a identificação hash em cima deste prefixo. De certo modo, o prefixo gera entropia suficiente para fazer os processos de amostragem e etiquetamento parecem aleatórios.

Computou-se o número de colisões em um milhões de pacotes, como uma função do tamanho do prefixo do pacote; veja figura 4. Está claro que só confiando no cabeçalho do pacote não é suficiente para a amostragem de trajetória trabalhar bem, como cabeçalhos idênticos aparecem muito freqüentemente (l = 20 bytes). Porém, aumentando o tamanho do prefixo do pacote para levar em conta alguns bytes do payload rapidamente diminui a probabilidade de colisão. Aumentando o tamanho do prefixo do pacote além de 40 bytes aproximadamente, as colisões não reduzem mais; as colisões restantes são devido a pacotes

que realmente são cópias exatas de ao menos um outro pacote. Porém, as colisões são suficientemente raras ao ponto de serem inconseqüentes.

2.7 Avaliação de Funções Hash

Foi explorado as propriedades estatísticas de algoritmos hashing em rastreamento dos pacotes[ND]. As rotas foram juntadas usando a utilidade *tcpdump* [16] em um host preso a um segmento de rede da área local perto da borda de uma rede do campus. A análise foi executada em quatro rotas cada uma incluindo um milhão de pacotes IP[ND]. As funções hash foram implementadas em aritmética de inteiro de 32 bits através da divisão longa por palavras de 16 bits. Uma propriedade desejável da amostragem hash é que o pacote amostrado deva aparecer independente de um subconjunto do conteúdo do pacote. Por conseguinte, a distribuição de qualquer atributo variável do pacote (como endereço IP de fonte ou destino) deva ser o mesmo para pacotes amostrados como para a população original. Testes da hipótese de independência agora foram executadas baseados em estatísticas chi-quadradas calculadas das rotas amostradas e originais. Foram aplicados três variantes deste procedimento para testar a hipótese de independência.

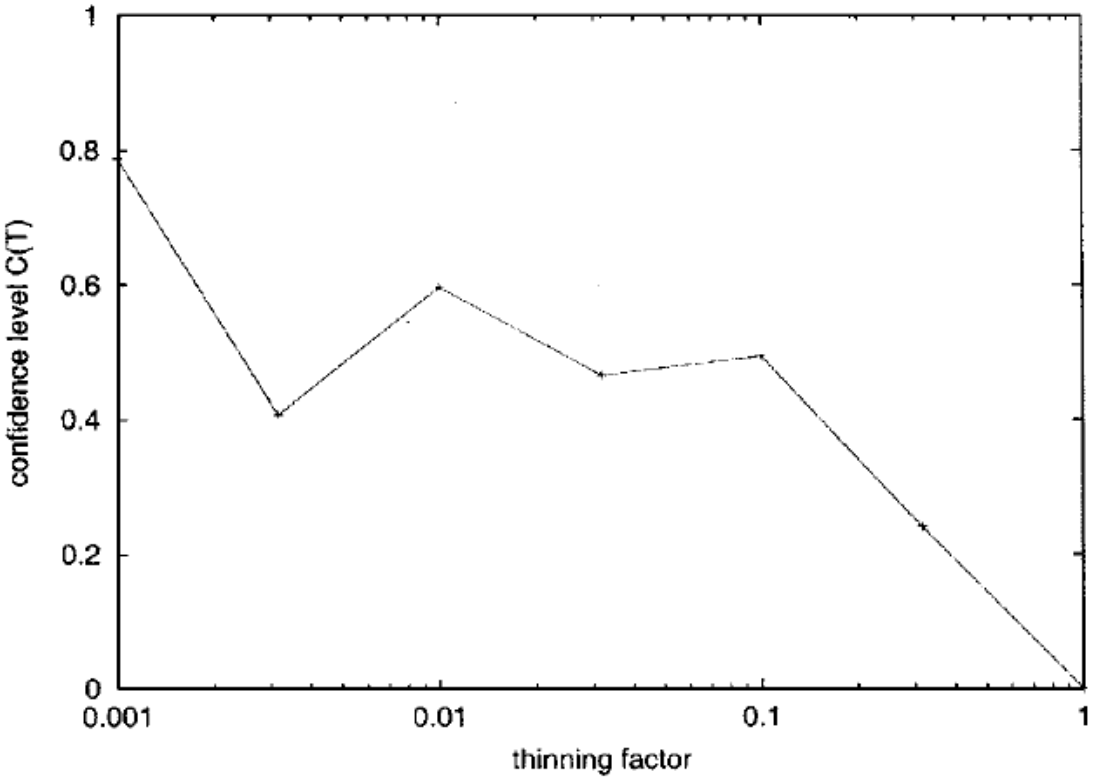

Figura 5. Amostragem Hash das distribuições de endereço. Níveis de confiança das estatísticas chi-quadradas de distribuições de endereço amostradas como uma função do fator de estreitamento. Em todos os casos, a distribuição das amostras é consistente com a rota cheia até 80% do nível de confiança. Amostragem hash é calculada em prefixo de pacotes de 40-bytes [ND].

Distribuições do Prefixo de endereço: O prefixo de endereço dos pacotes são analisados. A amostragem hash foi calculada usando 40 bytes do prefixo de pacote. Aumentando o

prefixo de pacote além deste ponto não diminui a freqüência de colisões (veja Figura 4), assim não é esperado nenhuma redução adicional em dependência entre amostragem hash e endereço do pacote.

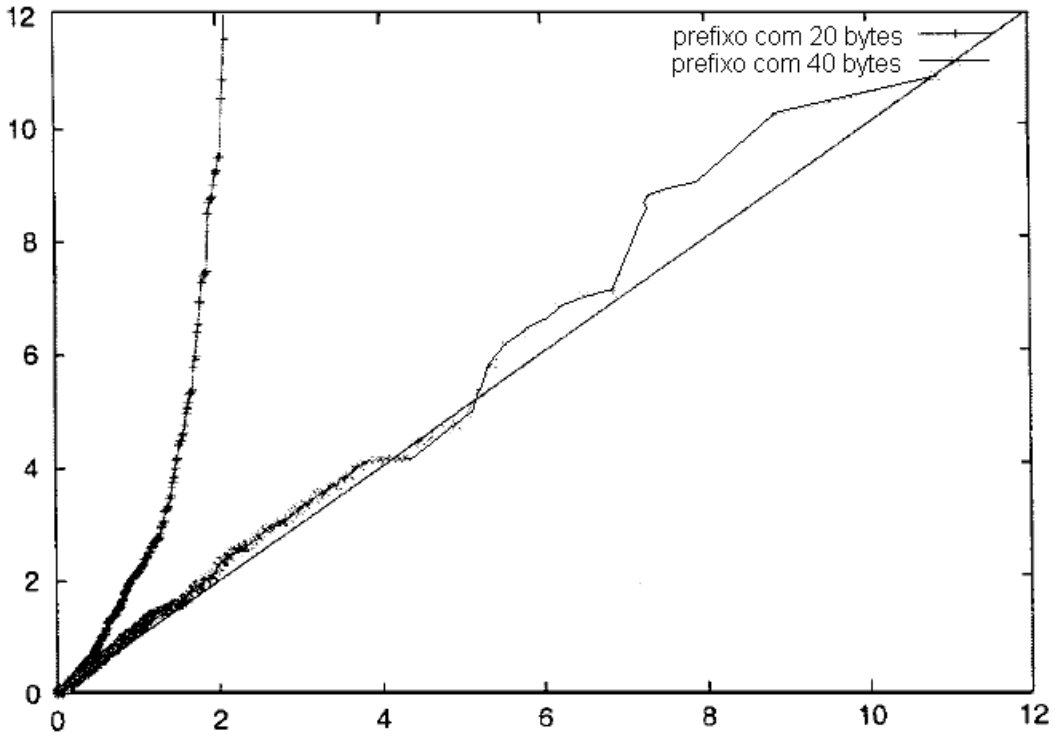

Figura 6. Amostragem Hash das distribuições de bits de endereço. Quantile-quantile grafico dos valores dos bits de endereço da distribuição chi-quadrada versus a distribuição chi-quadrada com um grau de liberdade; para várias rotas [ND]

Distribuições dos bits de endereço: Para cada bit de posição é construído uma tabela 2-por-2 de contingência na qual os elementos são função do número dos pacotes e da função de amostragem. É calculado a correspondente estatística chi-quadrada T para cada bit do endereço. De acordo com a hipótese nula, cada qual deveria seguir uma distribuição chi-quadrada com um grau de liberdade. Na figura 6 estas estatísticas estão resumidas. Mostra-se a concordância dos dados; o gráfico é semelhante aquele obtido usando estatísticas aleatoriamente geradas da distribuição esperada.
Distribuições de amostragem temporais: Para uma rota de uma única sessão de ftp entre dois hosts, o processo de amostragem de pacote é consistente com a amostragem independente a uma taxa de amostragem media.

3 AMOSTRAGEM ÓTIMA

Nesta seção foi discutida a escolha do número de amostras e o número de bits por amostra. Por conveniência, denotamos $M=2^m$ como o tamanho do alfabeto da hash de identificação. Obviamente enfrenta-se duas metas contraditórias para a escolha de *n* e *m*. Por um lado, a confiabilidade das estimativas de tráfego aumenta com o número de amostras não

ambiguas que podemos coletar. Por outro lado, temos que limitar a quantia total de tráfego de medida entre os roteadores no domínio e no sistema de coleção. Note que a quantia de tráfego medido em um período de medida é determinado por mn bits, porque uma etiqueta de m-bit é transmitida ao sistema de coleção para cada uma das n amostras (ignorando headers de pacote para os pacotes de medida e outros overheads).

Foi formulado o simples problema de otimização seguinte: queremos maximizar o número esperado de amostras únicas (não ambiguas), sujeito a limitação de que o total de tráfego de medida não deve exceder uma constante c predefinida[ND]. Foi assumido que cada amostra independentemente leva um dos M valores de etiquetas com probabilidade uniforme $p = 1 / M$. A distribuição marginal do número de amostras que levam uma determinada etiqueta é o binômio B(n,p). Conseqüentemente, a probabilidade que a etiqueta é gerada precisamente uma vez no domínio com o período de medida é :

$$p_u = np(1-p)^{n-1} \quad (7)$$

O número médio de amostras únicas U(m,n), fixando através de m=c/n é dado por

$$U(n) = n(1-2^{-c/n})^{n-1} \quad (8)$$

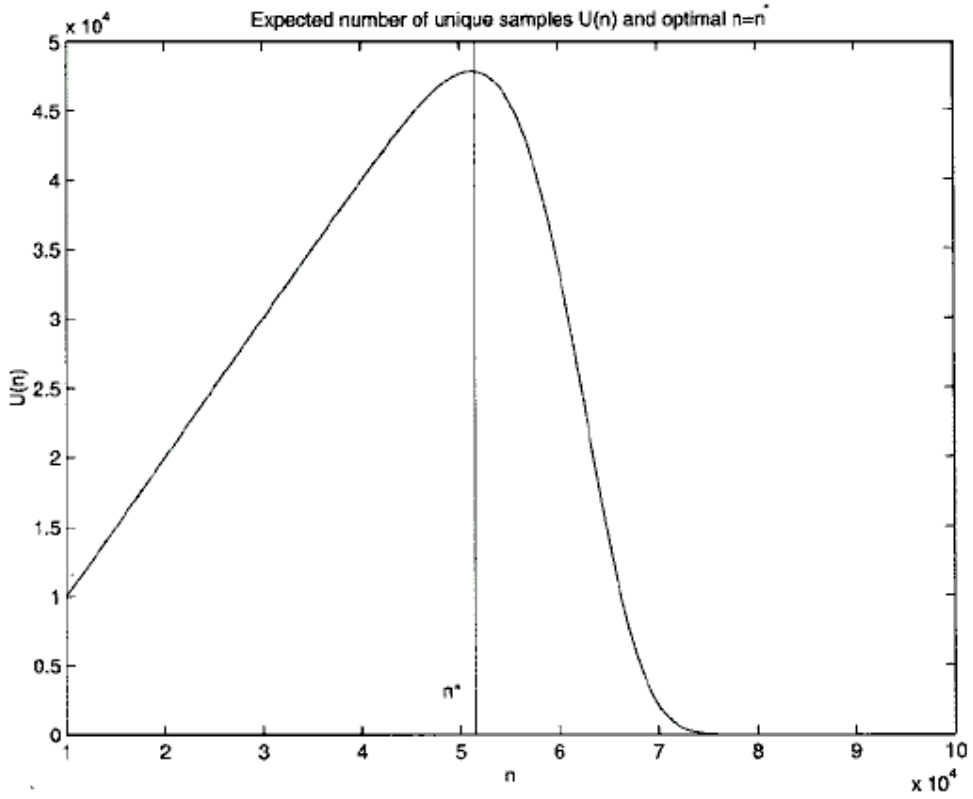

Figura 7. o número esperado de única amostra U(n) como uma função de n, para $c = 10^6$ bits. O ótimo número de amostras n é aproximadamente 5.15×10^4, com m = 19.4 bits por etiqueta. A probabilidade de colisão p é aproximadamente 0.072, isto é, 7.2% das amostras transmitidas ao sistema de coleção têm que ser descartadas[ND].

Da análise, podemos aproximar o ótimo número de amostras para grandes valores de M.

$$n^* \sim M^* / \log M^* \quad (9)$$

Esta aproximação é boa até mesmo para M* só moderadamente grande,

devido a natureza de variação lenta da função de logaritmo.
Embora foi definido M* = *c* log 2, a notação é consistente em que M* é, assintoticamente, o tamanho do alfabeto usado para representar o número maximizado de amostras únicas. Isto é porque o tamanho de etiqueta correspondente é o mais baixo inteiro maior que ou igual a $m^* = c / n^* \sim c \log M^* / M$
Foi computado a probabilidade de colisão de amostragem no ponto operacional ótimo.

$$p_{coll} = 1\text{-}e^{-1/(m^*\log 2)} \quad (10)$$

A figura 7 ilustra como n = n* maximiza U(n) : para n < n*, as colisões são muito raras e desperdiçamos bits de etiqueta para muito poucas amostras; para n > n*, as colisões são muito freqüentes e desperdiçamos amostras com colisões porque identificadores de etiqueta são muito pequenos. Note que obviamente o ponto ótimo não pode ser alcançado exatamente. Na pratica, escolhe-se o maior inteiro B ≤ M* que satisfaz as condições.

4 MEDIDA DE TRÁFEGO

Nesta seção, foi usado amostragem de trajetória para uma tarefa de medida simples. A meta desta experiência é ilustrar como podem ser construídos estimadores baseados nas etiquetas amostradas e recebidas do domínio de medida. Assuma que um provedor de serviço quer determinar que fração de pacotes em um certo link do back bone pertence a um certo cliente (conforme Figura. 8). Para calcular esta fração, o provedor de serviço pode confiar nas etiquetas coletadas do link do back bone e do link de acesso onde o cliente conecta-se à rede.

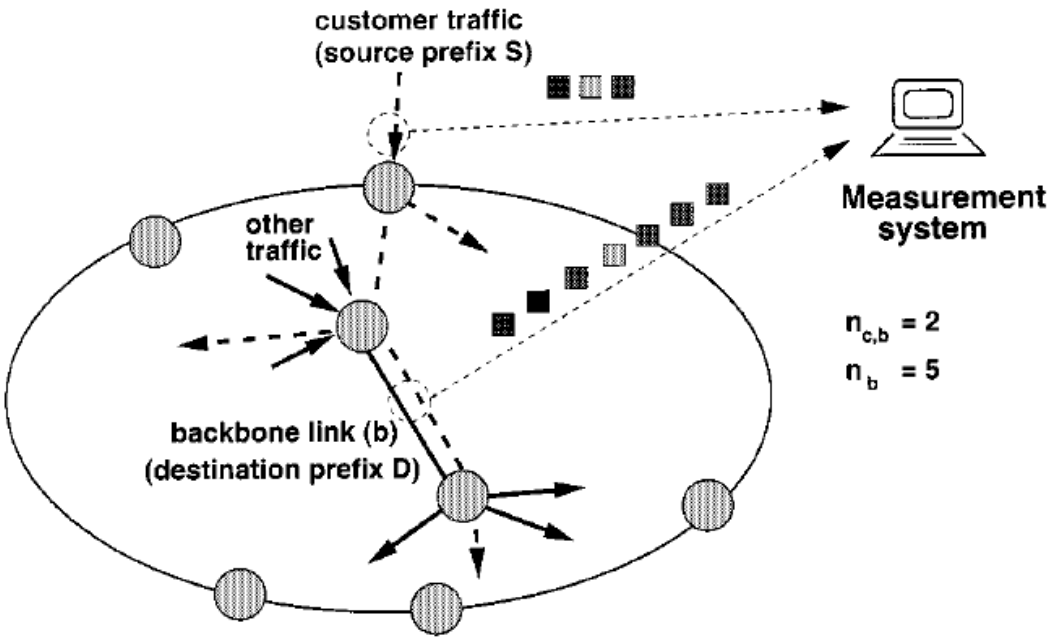

Figura 8. Experiência de medida. Uma experiência simples onde a etiqueta de dois links são comparadas para calcular que fração de tráfego no link do back bone vem do link de acesso do cliente [ND].

Com a finalidade de experimentação, a rota do pacote usada na seção prévia foi adaptado ao contexto presente como segue. Todos os pacotes de fonte com um certo prefixo S dão designados como originando do cliente de referência, enquanto os pacotes restantes são assumidos serem tráfego de fundo de outras fontes. Semelhantemente, só pacotes com um certo prefixo de destino D são assumidos que cruzam o link do back bone, enquanto o outros pacotes restantes não são roteados por este link. Para esta experiência, os detalhes do topologia não interessam. Para calcular a medida de interesse das etiquetas coletadas, procede-se assim: qualquer etiqueta que aparece mais que uma vez no domínio de medida inteiro é discartada. Entre as etiquetas únicas restantes, determina-se quais etiquetas só são observado no link do back bone, e quais etiquetas são observadas em ambos os links. Isto nos permite obter uma estimativa para a fração do tráfego do cliente no link do back bone, dado por

$$\mu = \frac{n_{cb}}{n_b} \tag{11}$$

onde $n_{c,b}$ é o número de etiquetas únicas observadas em ambos os links de acesso do cliente e no link do back bone, enquanto n_b é o número total de etiquetas únicas observadas no link do back bone.

Figuras 9 e 10 comparam as frações de trafico calculado e real no link do back bone, para dez períodos de medidas sucessivas. Para simplicidade, definiu-se um período de medida como uma sucessão de 10^3 pacotes sucessivos, no lugar de um intervalo de tempo. O gráfico também mostra intervalos de confiança ao redor dos valores calculados. Os intervalos de confiança são obtidos como segue. Computando o desvio padrão do estimador μ que assume que cada pacote é amostrado independentemente e com probabilidade igual. Se isto é verdade, então a probabilidade que um pacote amostrado pertence ao cliente é μ . O desvio padrão do estimator é então:

$$\sigma = \sqrt{\frac{\mu(1-\mu)}{n_b}} \tag{12}$$

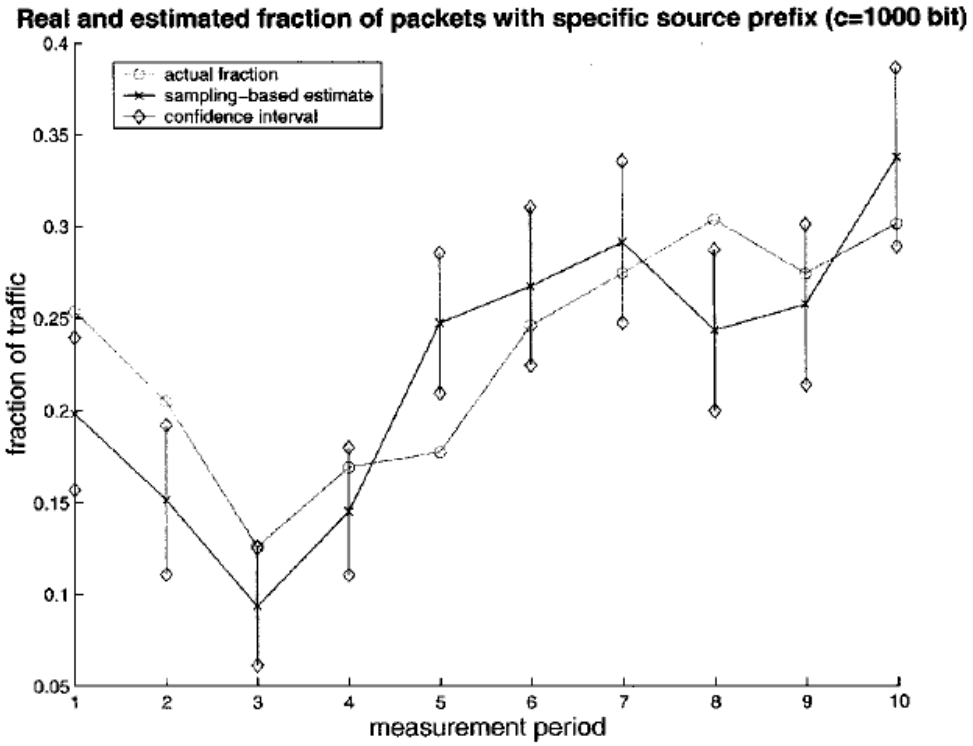

Figura 9. Fração de tráfego real e calculada do cliente. Para c = 1000 bits para este link (M = 693.1, B = 691, n = 106) [ND].

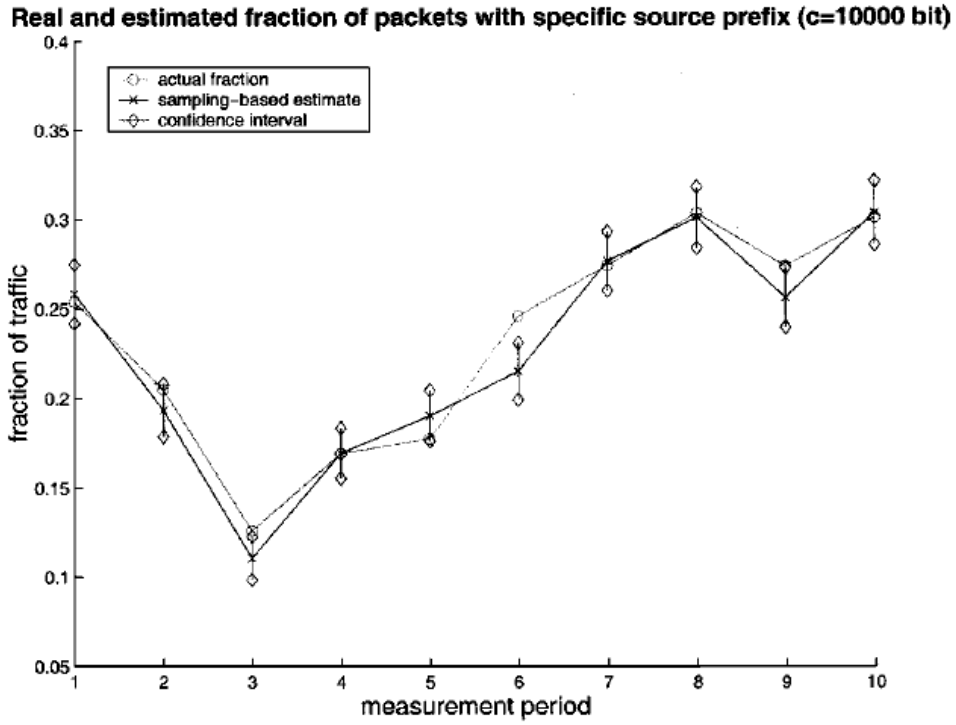

Figura 10. Fração de tráfego real e calculada do cliente. Para c = 10 kbit para este link (M = 6931.5, B = 6917, n = 782) [ND].

O intervalo de confiança que foi plotado é um desvio padrão ao redor do valor calculado. Note que a quantia de tráfego de medida por período de medida do link do back bone é bastante pequeno (1000 bits na Figura 9 e 10 kb na Figura 10). O intervalo de confiança é reduzido a medida que aumentamos o tráfego de medida.
Um estimator estatístico como o considerado aqui confia em um processo de amostragem randomico. O tamanho do intervalo de confiança é então uma conseqüência do teorema do limite central para variáveis randômicas independentes. Porém, a amostragem de trajetória é baseada em um processo de amostragem deterministico, e a decisão de amostragem para um pacote é uma função do conteúdo deste pacote. Não obstante, observa-se nesta experiência que o verdadeiro valor da quantidade calculadas caem, sem exceção, dentro ou muito perto do intervalo de confiança. Isto ocorre apesar do fato que há correlação forte entre o conteúdo dos pacotes (porque todos os pacotes do cliente têm a mesmo prefixo de fonte) e os eventos que estamos contando (pacotes pertencentes ao cliente).
Esta correlação não traduz em um processo de amostragem influenciada aqui. Isto demonstra que boas funções de hash podem randomizar suficientemente as decisões de amostragem tal que o conjunto de pacotes amostrados (e as suas etiquetas) são representativos de todo o tráfego para finalidade de estimação estatística.

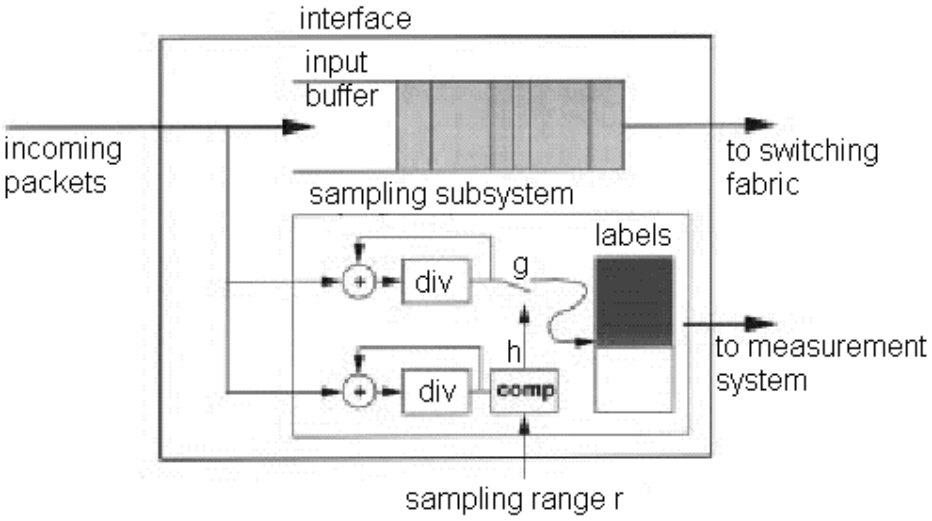

Figura 11. Implementação. Uma possível implementação de amostragem de trajetória computa tanto a amostragem hash quanto a identificação hash concorrentemente durante o fluxo. Isto remove a necessidade de se fazer uma cópia separada de cada pacote. A computação dos dois hashes, definidas em (7), pode ser implementada com a função elementar multiplicar-e-somar suportada pelos DSPs. Um pequeno buffer armazena as etiquetas antes delas serem copiados para um pacote IP e enviadas ao sistema de coleção. Alguma lógica adicional seria necessária em alguns nós (em nós de ingresso mais lentos) para extrair outros campos de interesse de um pacote, por exemplo, tamanho, endereços de fonte e de destino [ND].

5 DISCUSSÃO

5.1 Implementação

Foi discutido que o custo de implementação da amostragem de trajetória é até mesmo bastante aceitável para a interface de mais alta velocidade disponível hoje[ND]. Amostragem de trajetória requer um dispositivo para cada interface capaz de: 1) computar a amostragem hash e fazer uma decisão de amostragem e 2) computar a hash identificação para os pacotes amostrados.

O custo computacional é dominado obviamente pelas operações que têm que ser executado para cada pacote que atravessa esta interface (ao invés de operações só em pacotes amostrados). Na descrição conceitual do processo de amostragem, tem-se a computação da amostragem e a hash identificação como sequenciais. A hash de identificação só seria computada se o pacote fosse amostrado, caso contrário o pacote é descartado. Porém, de um ponto de vista de implementação, isto é indesejável, como isto requereria armazenamento para cada pacote até que o hash amostrado fosse computado.

Um implementação alternativa ilustrada na Figura 11 computa tanto o hash de amostragem quanto a hash de identificação.

As "hash function" já discutida permite tal implementação. Isto remove o fardo de ter que fazer uma cópia separada do pacote com a finalidade de computar a hash de identificação. O processador computa ambos os hash, e simplesmente escreve o hash de identificação *g* na etiqueta de armazenamento se o hash de amostragem *h* é igual a um. A etiqueta de armazenamento acumula etiquetas de pacotes até que alcança um tamanho predefinido, então envia as etiquetas para o sistema de medida como um único pacote IP.

Como um exemplo, uma DSP estado-da-arte pode processar até aproximadamente 600M 32-bit [multiplicar-e-acumular (MAC)] operações por segundo. Isto corresponde a uma taxa de fluxo de dados de 20 Gbit/s. Além disto, a largura de banda do fluxo na memoria I/O pode chegar a 256 bit por ciclo de memoria, o que corresponde a 77 Gbit/s a um clock de 300-MHz. Em comparação, uma interface OC-192 (a interface SONET mais rápida disponível comercialmente) alcança 10 Gbit/s.

Enquanto estes argumentos estão baseado em desempenho de processador de ponta, que não pode ser sustentado tipicamente por várias razões, estes números ilustram que as exigências computacionais necessárias para a amostragem de trajetória estão ao alcance de processadores atuais. Também é interessante notar que o preço de tal processador é duas ordens de magnitude abaixo que uma placa de interface OC-192. Adicionando lógica para amostragem de trajetória para interfaces de alta velocidade seria então comparativamente baratas. Também note que para adicionar apoio de medida para placas de interface está em linha com a tendência dos últimos anos para moverem o poder de processamento e funcionalidade do roteador para as interfaces.

Espera-se que o custo relativo da amostragem lógica com respeito ao hardware de interface evoluir em favor do método. De fato, parece que desempenho de processador aumenta ligeiramente mais rapidamente (dobrando em 18 meses de acordo com a lei de Moore) que velocidade de tronco de máximo (dobrando em 21 meses) [23].

Se estas tendências persistem, então o custo de incorporar amostragem de trajetória nas próximas gerações de interfaces de alta velocidade podem tornar-se desprezível.

O dispositivo de amostragem de link também requer uma interface de administração simples para habilitar/desabilitar amostragem de pacote, para dizer ao dispositivo onde enviar o tráfego de medida, e fixar os parâmetros das "hash function". Um protocolo de

administração de rede simples (SNMP) base de informação de administração (MIB), indexado pelo endereço IP da interface, poderia cumprir esta função.

5.2 Comparação com Outras Aproximações

Foi discutido várias aproximações de medida comuns para redes de IP e os pusemos em perspectiva levando em conta os pontos importantes [ND]. Há duas classes gerais de aproximações de medida. Aproximações baseadas em agregação são funções deterministicas dos dados observados. Elas normalmente computam a soma ou o máximo de alguma medida sobre um conjunto de dados (por exemplo, a soma de pacotes que atravessam um link durante um intervalo, ou o máximo atraso fim-a-fim de ida-e-volta para um conjunto de pacotes). Aproximações baseadas em amostragem extrai um subconjunto aleatório de todas as possíveis observações. Este subconjunto de amostragem é suposto ser representativo do todo. A lei de grande números afirmam aquele estimadores metricos fidedignos desejados pode ser construído destas amostras.

Medidas de Link (baseado em agregação, Direto): Nesta aproximação, estatísticas de tráfego agregado são medidas em um link base, e é informado periodicamente (por exemplo, a cada cinco minutos). Medidas tipicamente incluem o número de bytes e pacotes transferido e roteados dentro de um período informando. Algumas destas estatísticas são definidas como parte do SNMP MIBs [27]. A limitação desta aproximação é que um pouco de informação é perdido na agregação; então, não permite classificar o tráfego (por exemplo, por tipo protocolar, fonte ou endereço de destino, etc.). Mais importante, não é em geral possível deduzir fluxo espacial de tráfego, isto é, deduzir que caminhos o tráfego segue entre um ponto de ingresso e um de egresso. Como tal, esta aproximação é melhor para descobrir problemas potenciais, que se manifestam por congestionamento de link, do que analisar o problema de fato e modificar as informação de roteamento para remediar isto.

Agregação de fluxo (baseado em agregação, Indireto): Nesta aproximação, um ou vários roteadores dentro do domínio coletam medidas por fluxo. Um fluxo inclui uma sucessão de pacotes com alguns campos comuns em seus cabeçalhos e que são agrupados a tempo [8], [19]. O roteador tem que manter um cache de fluxos ativos. Um registro de fluxo pode incluir especificação do endereço IP da fonte e destino e número da porta, tempo de inicio do fluxo, duração, o número de bytes e pacotes, entre outros. Uma desvantagem da agregação de fluxo é que a quantia de dados de medida podem ser consideráveis; o tráfego gerado pode impor uma carga adicional significante na rede. Isto é especialmente verdade na presença de números grandes de fluxos pequenos, como http-get pedidos. O tráfego de medida é difícil de predizer. Depende pesadamente do modo que o roteador identifica fluxos individuais que dependem de vários parâmetros controle (como o grau de agregação do endereço de fonte e destino), a mistura de tráfego (protocolos), e o tamanho do cache. Uma complicação adicional pode surgir se medidas de tráfego forem usadas para funções de controle de tempo real. Desde que um fluxo registrado normalmente é gerado só na conclusão de um fluxo, isto implica que uma estatística on-line pode perder um fluxo duradouro que ainda não tenha terminado.

Uma matriz de caminho completa sobre o domínio pode ser obtida se medidas de agregação de fluxo estão disponíveis em cada ponto de ingresso e se sabemos como o tráfego é roteado pelo domínio. Enquanto esta é atualmente a única aproximação para obter uma matriz de tráfego completa em redes IP, tem várias desvantagens:

- Emulação dos protocolos de roteamento: Até mesmo para roteamento não adaptativo, temos que confiar em emulação do protocolo de roteamento para corretamente mapear as

medidas do trafico de ingresso sobre a topologia da rede; isto requer conhecimento completo dos detalhes do protocolo de roteamento como também sua configuração.
- Nenhuma verificação: Como mencionado antes, um papel importante, de medida de tráfego está na verificação e resolução de probemas de protocolos e políticas; obviamente, emulação de roteamento impede detectar problemas reais no roteamento, por exemplo, devido a bugs no protocolo.
- Roteamento dinâmico e adaptativo: Roteamento dinâmico (roteado ao redor de links falhos) ou roteamento adaptativo (balanceamento de carga por links/paths múltiplos) complicam emulação, porque informação do estado precisa do link teria que estar disponível em cada momento (note que protocolos de roteamento extensamente usados como OSPF têm algumas providências para equilibrar carga entre vários caminhos menores em um modo pseudorandomico; seria impossível emular isto exatamente).

Sondas Fim-a-fim ativas (baseado em amostragem, Indireto): Nesta aproximação, hosts (pontos finais) conectados à rede envia pacotes de sonda a um ou vários outros hosts para calcular medidas do caminho, como a taxa de perda de pacote e o atraso de ida-e-volta, [5], [1], [2]. Em uma variação desta aproximação, hosts não geram de fato pacotes de sonda, mas eles coletam e trocam medidas do tráfego de uma sessão de multicast (por exemplo, RTCP [25]).
Esta aproximação dá medidas diretas de caminho de características fim-a-fim, como demora de ida-e-volta e taxa de perda de pacote; características de link têm que ser deduzidas. Esta aproximação pode ser vista como um modo alternativo para obter medidas agregado por link. Sua vantagem é que não requer nenhuma medida de apoio da rede. Tem as mesmas desvantagens da aproximação “medida de link”.

6 CONCLUSÃO

No artigo em análise foi proposto um método para a consistente amostragem de trajetórias de pacote em uma rede. A amostragem seleciona um subconjunto de pacotes, mas se um pacote é selecionado em um link, também será selecionado em todos os outros links que atravessa. Ao atravessar a rede, cada pacote implicitamente indica se deveria ou não ser amostrado através de sua parte invariante, isto é, aqueles bits que não mudam de link para link. Um hash destes bits é calculado em cada roteador, e só aqueles pacotes cujo amostragem hash caem dentro de uma determinada gama de valores são selecionados. Para pacotes selecionados, um hash diferente, o hash de identificação, é usado para estampar uma identidade no pacote. Isto é comunicado pelo roteador de amostragem para os sistemas de medida. Isto habilita posterior análise amostral das trajetórias distintas uma vez que as amostras são reportadas. O método tem várias propriedades desejáveis:

- Processo Simples: As únicas operações por pacote requeridas são a aritméticas de divisão de um número pequeno de bytes no cabeçalhode pacote. Nenhuma classificação de pacote ou busca em memória são usados.
- Nenhum Estado do Roteador é requerido e pacotes são processados individualmente. Nenhum memória cache é requerido no subsistema de medida do roteador, evitando atraso de cache e possível influenciamento pela exigência de políticas de expiramento de cache. Isto não exclue a possibilidade de ter um estado no sistema de informação no roteador; pode ser desejável para relatórios discretos para o sistema de medida em lugar de os enviar individualmente.
- Pacotes são observados diretamente: O curso dos pacotes pela rede pode ser determinado sem um modelo de rede que especifica como eles devem ser roteados. Isto é importante para depuramento, desde que roteamento pode não especifiar facilmente o corrente estado de roteamento do sistema [15],

[33]. Além disso, configuração ou outros erros podem causar comportamento no roteamento que divirja daquele especificado pelo modelo. No futuro, é planejado investigar o desempenho de uma classe mais larga das “hash function”. Há duas motivações aqui. Primeiro, deseja-se avaliar a utilidade das “hash function” com propriedades randomicas mais fortes, por exemplo, “universal hashing” [29]. Segundo, há vantagens de implementação usando “hash function”, como MD5 [22], que já deve estar disponível em elementos de rede.

Também é proposto avaliar a amostragem de trajetória em um contexto de rede. Os objetivos são entender o reportamento da trajetória em cima de uma grande rede, e desenvolver técnicas para sistemática reconstrução de trajetórias, inclusive resolução de ambigüidades. A aproximação combina informação de roteamento e rotas de tráfego para fazer uma simulação da rede que captura a topologia e padrões de tráfego de reais redes.

7 REFERÊNCIAS BIBLIOGRÁFICAS UTILIZADAS NO ARTIGO

[1] G. Almes, S. Kalidindi, and M. Zekauskas. (1999, Sept.) A one-way delay metric for IPPM. [Online]. Available: http://www.ietf.org/rfc

[2] , (1999, Sept.) A one-way packet loss metric for IPPM. [Online]. Available: http://www.ietf.org/rfc

[3] D. O. Awduche, "MPLS and traffic engineering in IP networks," *IEEE Commun. Mag*., vol. 37, pp. 42–47, Dec. 1999.

[4] Y. Bernet, J. Binder, S. Blake, M. Carlson, B. E. Carpenter, S. Keshav, E. David, B. Ohlman, D. Verma, Z. Whang, and W. Weiss. (1999, Feb.) A framework for differentiated services. [Online]Internet draft.

[5] R. Cáceres, N. G. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal loss characteristics," *IEEE Trans. Inform. Theor*y, vol. 45, pp. 2462–2480, 1999.

[6] R. Callon *et al*.. (1999, Sept.) A framework for MPLS. [Online]Internet draft.

[7] CERT/CC and FedCIRC, "Advisory CA-2000-01 denial-of-service developments," Carnegie Mellon Software Engineering Institute, Jan. 2000.

[8] K. C. Claffy, H.-W. Braun, and G. C. Polyzos, "Parameterizable method-ology for Internet traffic flowprofiling," *IEEE J. Select. Areas Commun*., vol. 13, pp. 1481–1494, Oct. 1995.

[9] K. C. Claffy, G. C. Polyzos, and H.-W. Braun, "Application of sampling methodologies to network traffic characterization," *Comput. Commun. Rev*., vol. 23, no. 4, pp. 194–203, Oct. 1993.

[10] I. Cozzani and S. Giordano, "Traffic sampling methods for end-to-end QoS evaluation in large heterogeneous networks," *Comput. Networks and ISDN Syst*., vol. 30, pp. 16–18, Sept. 1998.

[11] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford, "NetScope: Traffic engineering for IP networks," *IEEE Network Mag*., vol. 14, pp. 11–19, Mar. 2000.

[12] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, "Deriving traffic demands for operational IP networks: Method-ology and experience," in *Proc. ACM SIGCOM*M, Stockholm, Sweden, Aug. 2000, pp. 257–270.

[13] S. Floyd and V. Jacobson, "Random early detection gateways for con-gestion avoidance," *IEEE/ACM Trans. Networkin*g, vol. 1, pp. 397–413, Aug. 1993.

[14] S. Floyd and V. Jacobson, "The synchronization of periodic routing mes-sages," *IEEE/ACM Trans. Networkin*g, vol. 2, pp. 122–136, Apr. 1994.

[15] T. G. Griffin and G. Wilfong, "An analysis of BGP convergence proper-ties," *Comput. Commun. Rev*., vol. 29, pp. 277–288, Oct. 1999.

[16] V. Jacobson, C. Leres, and S. McCanne. (1989) tcpdump. [Online]. Available: ftp://ftp.ee.lbl.gov

[17] D. E. Knuth, "Sorting and searching," in *The Art of Computer Program-min*g. Reading, MA: Addison-Wesley, 1998, vol. 3.

[18] C. Labovitz, G. R. Malan, and F. Jahanian, "Internet routing instability," *IEEE/ACM Trans. Networkin*g, vol. 6, pp. 515–528, Oct. 1998.

[19] NetFlow, Cisco Systems. (2000). [Online]. Available: http://www.cisco.com/warp/public/732/netflow/index.html

[20] V. Paxson, "Empirically derived analytic models of wide-area TCP con-nections," *IEEE/ACM Trans. Networkin*g, vol. 2, pp. 8–17, Aug. 1994.

[21] K. K. Ramakrishnan and S. Floyd. (1999, Jan.) A proposal to add explicit congestion notification (ECN) to IP. [Online]. Available: ftp://ftp.isi.edu/in-notes/rfc2481.txt
[22] R. Rivest. (1992, Apr.) The MD5 message-digest algorithm. [Online]. Available: ftp://ftp.isi.edu/in-notes/rfc1321.txt
[23] L. G. Roberts, "Beyond Moore's law: Internet growth trends," *IEEE Comput*., vol. 33, pp. 117–119, Jan. 2000.
[24] L. Sachs, *Applied Statistic*s. New York: Springer, 1984.
[25] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. (1996, Jan.) RTP: A transport protocol for real-time applications. [Online]. Avail-able: http://www.ietf.org/rfc
[26] W. Simpson. (1995) IP in IP tunneling. [Online]. Available: RFC 1853, ftp://ftp.isi.edu/in-notes/rfc1853.txt
[27] W. Stallings, *SNMP, SNMP v2, SNMP v3, and RMON 1 and* 2, 3rd ed. Reading, MA: Addison-Wesley, 1999.
[28] D. Tappan *et al*.. (2000, July) MPLS label stack encoding. [Online]In-ternet draft
[29] M. Thorup, "Even strongly universal hashing is pretty fast," in *Proc. 11th ACM-SIAM Symp. Discrete Algorithms (SODA*), 2000, pp. 496–497.
[30] V. Paxson and S. Floyd, "Why we don't know how to simulate the In-ternet," in *Proc. 1997 Winter Simulation Conf*., Dec. 1997.
[31] J. Walrand, private communication, Feb. 2000.
[32] K. C. Claffy, G. C. Polyzos, and H.-W. Braun, "Application of sampling methodologies to network traffic characterization," in *Proc. ACM SIG-COM*M, Sept. 1993, pp. 194–203.
[33] T. G. Griffin and G. Wilfong, "An analysis of BGP convergence proper-ties," in *Proc. ACM SIGCOM*M, Aug./Sept. 1999, pp. 277–288.

8 BIBLIOGRAFIA

[ND] N. G. Duffield, "Trajectory Sampling for Direct Traffic Observation" *IEEE Transaction on Networking*., vol. 9, pp. 280–292, June. 2001.
[AT] A. S. Tanembaum, "Redes de Computadores" *Editora Campus*, 3^0 Edição,1997, pp. 471–474.
[WW] http://logi.cc/linux/ipchains-log-format.php3 data 22/10/01