Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks

Grupo 1
Francisco Caeiro, **47823**
Bruno Andrade, **47829**
António Estriga, **47839**

Qual é o problema que os autores tentam resolver?

As redes celulares, ao contrário das redes tradicionais por fio ou das LANs *wireless*, são bastante variáveis. A velocidade de ligação que estas redes providenciam estão constantemente a variar, mesmo em questões de segundos. Como resultado, aplicações interativas, que dependam de velocidades constantes, acabam por sofrer.

Os protocolos de transportes lidam com estas variações de uma maneira reativa: se, ao enviar pacotes a uma certa velocidade, estes estão a ser bem recebidos, aumentam a velocidade; se a ligação sofrer uma variação inesperada (que acontece muito em redes celulares), os protocolos são lentos a diminuir a velocidade, *overshooting* o *throughput* da rede, causando assim uma acumulação de pacotes por entregar.

Tomando como exemplo as aplicações de videoconferência: ao existirem estas filas de pacotes, a interação é visivelmente interrompida, e pode demorar vários segundos até que os pacotes sejam transmitidos a uma velocidade minimamente desejada.

Este problema é relevante?

Cada vez mais pessoas acedem à Internet através de dispositivos móveis. Com o avançar tecnológico da sociedade, desejamos ser mais rápidos e melhores para que tenhamos mais tempo para fazer outras tarefas, muitas das vezes relacionadas com o nosso trabalho. Ao existir um protocolo que garante um bom *throughput* e um menor *delay time*, tendo em conta a variabilidade de capacidade das redes celulares, conseguimos melhorar a interatividade com, por exemplo, colegas de trabalho em conferências ou reuniões pela Internet.

Qual é a sua solução?

Este artigo apresenta um protocolo de transporte orientado a aplicações interativas em redes com velocidades variáveis denominado por Sprout. Sprout tem como objetivo oferecer às aplicações o maior throughput possível enquanto delimita o risco de *delay* acima de 100ms, prevenindo uma longa fila de espera de pacotes na rede.

Para isso, são realizados 3 grandes passos: 1. o algoritmo infere a velocidade de ligação a cada instante, em vez de a medir, porque sabemos que os redes celulares não são confiáveis e a sua velocidade vai mudar. Para a inferir, usa uma função de densidade de probabilidade. 2. Através desta incerta inferência, o recetor origina um *forecast* prevendo quantos pacotes é que pode receber no pior dos casos, delimitando o risco de um pacote ficar à espera mais tempo do que a tolerância do emissor em 5%. O receptor envia este *forecast* para o emissor *piggybacking it* nos pacotes que enviará. 3. Com o *forecast* recebido, o emissor estuda o número de bytes que pode transmitir enquanto garante que cada pacote tem 95% de probabilidade de sair da fila em menos de 100ms. Para o fazer, Sprout olha 100ms para o futuro do *forecast*, conta o número esperado de bytes a sair da fila nesse tempo e subtrai a ocupação estimada da fila de espera atual. Usando esta tática, Sprout combina *pacing* com controlo de fluxo baseado em janelas.

Que novas técnicas foram usadas?

Sprout, em vez de usar um controlo de congestão reativo tal como o TCP, faz uma "previsão" da qualidade da ligação, tendo em conta o tempo de chegada dos pacotes ao recetor. Ao inferir, usando a esquema apresentado na pergunta anterior, este consegue evitar o *over-buffering* e o *under-utilization*, lidando corretamente com as variações irregulares e quase aleatórias na velocidade da rede.

É de notar também que este protocolo apenas é possível porque as redes celulares oferece a cada um dos seus utilizadores uma "fatia" da rede, não havendo misturas entre utilizadores das ligações e, consequentemente, das filas de espera dos pacotes. Ao contrário de sistemas típicos de controlo de congestão, o Sprout é focado em adaptar as condições de ligações variáveis, não em adaptar o tráfego simultâneo ao mesmo *bottleneck*. Se as redes celulares não separassem os seus utilizadores por ligações únicas, o Sprout iria oferecer o mesmo *delay* que as técnicas já existentes.

Como é que se destaca de trabalhos anteriores?

Ao contrário de aplicações ou protocolos existentes, o Sprout não reage a sinais de congestão, tais como perdas de pacotes ou aumento de RTT. Este protocolo consegue calcular antecipadamente a capacidade da rede! Como é conservativo, caso a capacidade da rede sofra alguma alteração (é uma rede celular, logo vai sofrer), Sprout oferece menos *throughput*, mantendo o *delay* baixo, para que a interação através das aplicações não seja perdida. Em todos os estudos realizados, Sprout oferece o menor *delay* dos restantes protocolos e aplicações avaliados.

Mostraram ainda que o Sprout, mesmo sendo *end-to-end*, consegue competir em termos de desempenho, com o Cubic-over-CoDel, um protocolo que é implementado pelos nós intermediários da rede, mostrando que o *Active Queue Management* pode não ser necessário para este tipo de aplicações.

Quais são os pontos mais fortes deste artigo? E os seus pontos fracos?

Não só mostraram com grande detalhe como foi implementado o Sprout (e disponibilizaram-no publicamente) como também o compararam a protocolos e aplicações existentes através de *trace-route emulation*, de modo a avaliar cada método sob as mesmas condições variáveis. Após o estudo destas emulações, verificaram que, em média, tanto no *downlink* como no *uplink*, o Sprout consegue obter um maior *throughput* e um menor *delay* que os restantes.

Estudaram ainda quais os benefícios do forecasting do Sprout, criando uma versão simplificada chamada Sprout-EWMA. Este protocolo usa também o tempo de chegada dos pacotes mas, em vez de realizar uma inferência com esses dados, coloca-os numa média móvel exponencial ponderada, um tipo de média móvel usado para medir, por exemplo, risco financeiro. Comparando com os restantes métodos, este consegue obter um maior *throughput* que o Sprout em troca de *delay* um pouco maior. Estudaram também a variação do parâmetro de confiança do *forecast*, mostrando, novamente, que é possível alcançar um maior *throughput* em troca de mais *delay*.

Concluindo, como ponto positivo, o utilizador é que decide se prefere o Sprout ou outros métodos já existentes. É-nos apresentado este protocolo, com bons resultados para um específico tipo de aplicações e customizável até a um certo ponto, mas, no final das contas, todas as redes são diferentes.

Como seria uma extensão deste trabalho?

Tendo em conta os bons resultados da implementação deste protocolo, poderia ser efetuado um questionário por engenheiros de rede para saber se o Sprout valeria a pena ser implementado em redes reais e em que situações.