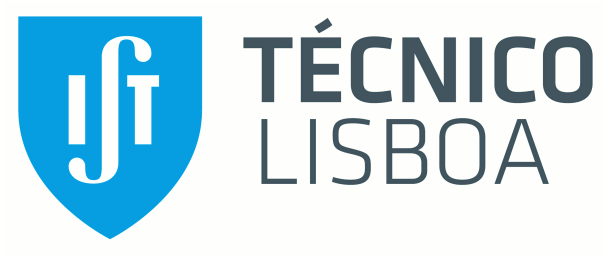

Algoritmia e Desempenho em Redes de Computadores

Inter-AS Routing and Path Inflation

Grupo Nº 14

Autores:

Diogo Rodrigues nº 84030
Pedro Moreira nº 84034

Professor:

João Luís Sobrinho

29/11/2019

1 Introdução

A Internet é um conjunto interconectado de cerca de 60 000 redes, denominadas *Autonomous Systems* (ASes). Este projeto tem como objetivo estudar e implementar algoritmos que realizem estatísticas do *Border Gateway Protocol* (BGP). O BGP é o protocolo de encaminhamento em execução entre as ASes da Internet. Este protocolo, com base nas relações comerciais entre as ASes, determina como o tráfego irá fluir na Internet e está configurado de forma a que cada Internet Service Provider (ISP) faça lucro do negócio de transitar tráfego de acordo com a relação comercial estabelecida com as ASes vizinhas.

Todo os algoritmos deste projeto foram implementados e testados tendo como base o grafo de ASes da Internet disponibilizado pelo docente na página da UC. O restante documento está organizado da seguinte forma: na secção 2 propõem-se algoritmos que verifiquem a validade das propriedades do grafo de não conter ciclos de clientes (2.1) e de estar comercialmente conexo (2.2); na secção 3 apresentam-se os algoritmos principais do projeto, nomeadamente, produzir estatísticas acerca do tipo de rotas comerciais eleitas por cada AS para chegar a uma AS de destino comum (3.1), produzir estatíticas do comprimento da rota comercial mais curta de cada AS para chegar a uma AS de destino comum (3.2) e comparar a mesma com o comprimento das rotas mais curtas sem quaiquer restrições comerciais (3.2.1); na secção 4 analisam-se os resultados obtidos e discutem-se os mesmos.

2 Propriedades do Grafo

A representação do grafo escolhida foi através de uma lista de adjacências, visto que embora a representação em matriz de adjacências fosse mais rápida, em termos de memória poderia não ser exequível. Foi então definido um vetor estático de estruturas em que cada índice irá identificar uma AS (foi definido um tamanho máximo para o número de ASes de $2^{16} = 65536$). A estrutura referida contém 3 listas que serão as listas de adjacências correspondentes aos vizinhos de cada AS que são, respetivamente, fornecedores, pares ou clientes. Registando cada AS desta forma consegue-se que quando se necessita de pesquisar pelas ASes vizinhas com uma determinada relação comercial com essa AS, essa pesquisa seja muito mais eficiente pois não existe necessidade de percorrer a lista pelas ASes que não interessam.

Outro aspeto importante é que logo na criação do grafo são identificados os índices no vetor que não correspondem a nenhuma AS (pois o número de ASes pode ser muito inferior à dimensão do vetor) e os índices que correspondem a *Tier-1* ASes.

2.1 Ciclo de Clientes

Para que o BGP atinja um estado estável único é necessário que não exista nenhum ciclo de clientes. Entenda-se por estado estável um estado final do BGP, ou seja, não há mensagens de encaminhamento BGP em trânsito na rede. Posto isto, é necessário perceber o que é um ciclo de clientes. Um ciclo de clientes é um ciclo fechado onde cada AS é cliente da próxima (caminhando sempre na mesma direção).

Para solucionar este problema utilizou-se uma depth-first search (DFS) em que se lançou uma DFS (de forma recursiva) de cada *Tier-1* AS, em que se teve em conta apenas as ligações referentes aos clientes. Desta forma, enquanto se estiver a descer em profundidade no grafo em direção a um *stub* (ASes sem clientes) se no "caminho" se visitar um nó não permitido, isto é, que já tenha sido visitado (no mesmo caminho) antes de se encontrar o *stub* final, deparamo-nos então com um ciclo de clientes. Para que fique mais explícito, uma AS está em avaliação (ou seja, não é permitida) quando é visitada, e é retirada de avaliação quando a função de DFS retornar depois de ter sido chamada para todos os seus vizinhos e os declarar como permitidos (pelo que assim *stubs* são sempre permitidos). Deste modo, consegue-se garantir que é detetada a presença de um ciclo de cliente caso este exista.

2.2 Comercialmente Conexo

Para definir que uma rede está comercialmente conexa, para cada AS de destino, todas as outras ASes elegem um tipo de rota (cliente, par ou fornecedor) para chegar ao destino em questão. Para verificar esta condição há uma condição relativamente simples, todas as *Tier-1* ASes estarem conectadas entre elas, pois desta forma ter-se-á abrangência total da rede. Assim, o algoritmo implementado consiste nisto mesmo, em pegar em cada *Tier-1* AS e verificar que a mesma está conectada a todas as outras.

3 Algoritmos Principais

Para calcular/contar as estatísticas indicadas na introdução deste relatório, foi necessário deli-

near as rotas estabelecidas pelas ASes, para cada tipo de preferências/restrições. Para tal, foi usado um algoritmo de procura em grafos, nomeadamente *breadth-first search* (BFS), com as adaptações necessárias para cada caso.

3.1 Encaminhamento de acordo com as relações comerciais

Para definir as rotas do "grafo da Internet", pelo interesse comercial de cada AS, foi utilizado o já conhecido algoritmo de *Dijkstra*. Correu-se o algoritmo partindo de cada AS e expandindo sempre todos os nós do grafo, como efeito de saber qual a rota escolhida por todas as ASes para a AS da qual se iniciou o *Dijkstra*. Desta forma, depois de correr o *Dijkstra* de todos os nós, ficaremos a saber a rota escolhida por todas as ASes para todas as ASes.

Visto que cada AS procura o seu benefício comercial, esta dará sempre prioridade a um envio por um cliente, e só em último caso fará um envio por um fornecedor. Pelo mesmo motivo (comercial), uma AS que receba uma mensagem de um fornecedor/par só aceitará reencaminhá-la para um cliente, por outro lado, caso a mensagem venha de um cliente, esta está elegível a ser reencaminhada por qualquer tipo de ligação. Graças ao facto da estrutura de dados utilizada para armazenar o grafo guardar as ligações de tipos diferentes em listas diferentes, o algoritmo só expandirá os nós das listas com tipos de ligações elegíveis.

Também, sendo que para efeitos comerciais o histórico da rota de um pacote a enviar não é relevante, à medida que se expande o *Dijkstra*, os custos das ligações já ultrapassadas não serão acumulados, visto que cada AS define a rota "pensando" exclusivamente em si.

Assim, definiu-se o custo de cada ligação entre ASes como a sua relação comercial: caso o próximo *hop* seja para um fornecedor do nó a ser expandido, o custo da ligação será 3; caso seja para um par o custo da ligação será 2; caso seja para um cliente o custo da ligação será 1. De notar que o *Dijkstra* aqui implementado, ao invés da implementação clássica do *Dijkstra*, está a determinar o tipo de rotas de todas as ASes para a AS de onde se iniciou o algoritmo (clássicamente seria o inverso), pelo que quando se refere que o próximo *hop* no *Dijkstra* seja para um fornecedor isto no fundo irá significar uma rota do tipo cliente, ao passo que quando o próximo *hop* seja para um cliente isto será o oposto, uma rota de fornecedor. Então, tendo em conta a forma como os custos foram definidos, o algoritmo dará prevalência a custos mais altos. Apesar de tradicionalmente o *Dijkstra* não se aplicar a problemas de maximização de custos, neste caso pode-se fazê-lo pois estes não são acumulados.

Para a fila de prioridades do *Dijkstra* foi utilizada uma *binary heap*, para a qual se alocam todos os elementos no início do programa e só se os libertam no final do mesmo, isto trará benefícios como ir-se-á ver em 4.

Correndo o *Dijkstra* com estas especificações, só serão exploradas ligações em que o próximo *hop* é para um cliente (equivalente a rota de fornecedor), depois de todas as ligações dos outros tipos (para pares e fornecedores) acessíveis no grafo já terem sido explorados (se o grafo for comercialmente conexo). Assim sendo, quando o objetivo é apenas obter os dados estatísticos relativamente aos tipos de rotas (por fornecedores, pares ou clientes), se o grafo for comercialmente conexo, força-se a paragem do algoritmo assim que é retirado um nó da *heap* correspondente a uma AS com uma ligação para um cliente (rota fornecedor), pois já se sabe que todas as outras serão do mesmo tipo.

3.2 Inflação de caminho

Para definir as rotas do "grafo da *internet*" não só pelo interesse comercial de cada AS, mas também pelos caminhos mais curtos possíveis (sendo que esta segunda condição nunca prevalece sobre a primeira) foi também utilizado o algoritmo de *Dijkstra* com o mesmo "core" que na secção 3.1, mas com uma alteração nos custos das ligações.

A condição de rota mais curta só deve ser considerada caso esta "desempate" duas opções com o mesmo interesse comercial e, uma vez que originalmente as comparações feitas eram entre custos (não acumuldados) de um valor do conjunto $\{1, 2, 3\}$, desta vez, passaremos a ter custos de $\{1, 2, 3\} \times NumberOfASes$ (onde $NumberOfASes$ representa o número total de ASes) subtraídos pelo número de ligações percorridas até então (equivalente ao número de *hops*). Como o número máximo de ligações percorridas será, no limite, o número total ASes existentes, o *gap* entre os pesos dos diferentes tipos de custos nunca será ultrapassado. O número de ligações já percorridas é subtraído pois o algoritmo utilizado dá prevalência a custos maiores, e desta forma, quanto mais longo for o caminho mais baixo será o custo, logo menos prioritário sê-lo-á.

3.2.1 Procura em largura sem restrições comerciais

De forma a poder tirar melhores conclusões foi implementado um algoritmo que produza a estatística dos caminhos mais curtos sem quaisquer restrições (não se teve em conta as relações comerciais entre as ASes) para que esta estatística possa ser comparada com a apresentada em 3.2. Para este ponto foi utilizado o já estudado algoritmo de *breadth-first search* (BFS) para cada AS da rede, pois não há melhor forma de fazer esta procura, garantido assim uma complexidade no pior caso de $\mathcal{O}(m+n)$, com m a ser o número de arestas (ligações) e n o número de vértices (ASes) no grafo. De realçar apenas que foi utilizada uma *queue* do tipo *FIFO* alocada estaticamente (máximo pré-estabelecido) para que desta forma seja o mais eficiente possível nos acessos, e que não tenha que alocar/libertar memória ao longo do processo.

4 Análise de Resultados

Ao longo do design e implementação dos algoritmos principais foram feitas otimizações que levaram a uma melhoria considerável da performance do programa e ir-se-á agora destacar algumas delas. O facto de em cada nó do grafo (cada AS) se guardar os tipos de ligações com as ASes vizinhas em listas diferentes permite ter uma procura muito mais eficiente, não fazendo comparações desnecessárias, o que levou a uma melhoria significativa na performance do programa. Outro detalhe de implementação que levou a uma maior velocidade de execução foi garantir que todos os elementos da *heap* são só alocados uma única vez ao longo do programa, uma vez que os elementos serão sempre os mesmos para todos o *Dijkstras* efetuados, uma repetitiva alocação e libertação dos elementos da fila de prioridades resultaria num aumento considerável do tempo de execução. Por último, caso o objetivo do programa seja somente obter estatísticas do tipo de rotas eleitas, ao interromper o *Dijkstra* quando é retirado um nó da *heap* correspondente a uma AS com uma ligação para um cliente (rota fornecedor), conforme explicado no final da secção 3.1, consegue-se uma abrupta melhoria na velocidade de execução - ao correr o programa com o ficheiro fornecido na página da UC consegue-se uma melhoria de 61%, este valor deve-se em muito ao facto da vasta maioria das rotas no grafo em questão (tal como na Internet) serem rotas de fornecedor, como confirmaremos de seguida.

Para o mesmo ficheiro mencionado, as estatísticas obtidas demonstram que 88.15% das rotas escolhidas são de fornecedor, 11.25% de pares e apenas 0.61% de clientes. Estes resultados vem de encontro com o que era expectável, uma vez que na Internet a maior parte das ASes são *stubs*, nomeadamente distribuidores de conteúdo, que não tem ASes clientes ou pares, pelo que terão de recorrer sempre a rotas de fornecedores. Observando agora os resultados obtidos nas figuras 1 e 2 observa-se que o número de hops utilizado, de modo geral, pelo caminho comercial mais curto é consideravelmente superior ao caminho mais curto sem restrições comerciais, o que é esperado dado que cada AS determina a direção para onde o tráfego flui pensando somente em si e de forma a ter lucro.

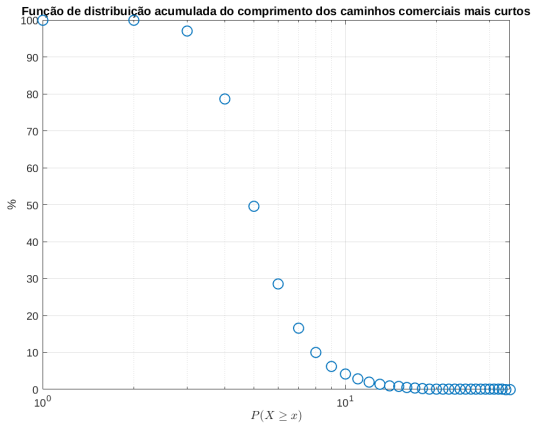

Figura 1: Função de distribuição acumulada do comprimento dos caminhos comerciais mais curtos (Nota: escala logarítmica no eixo dos *xx* para permitir uma melhor leitura do gráfico)

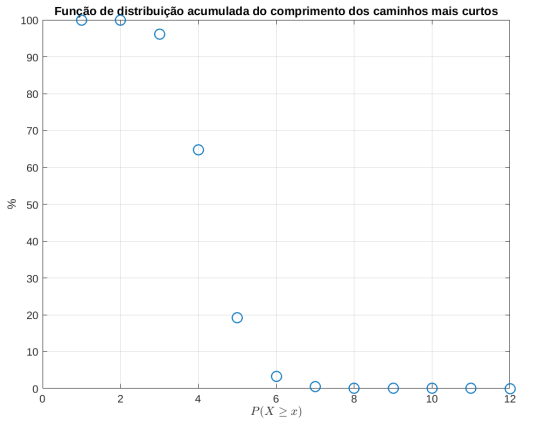

Figura 2: Função de distribuição acumulada do comprimento dos caminhos mais curtos