Grafos Evolutivos e Redes Tolerantes a Atrasos

César Gamboa Machado, Paulo Henrique Floriano
Orientador: Alfredo Goldman

Instituto de Matemática e Estatística da Universidade de São Paulo

Resumo

Atualmente, existem várias redes móveis com características dinâmicas. Nestes cenários, há grande intermitência nas conexões entre os nós da rede e protocolos de roteamento comuns não apresentam bons resultados.
Para resolver o problema do roteamento de pacotes em redes deste tipo, diversos algoritmos foram propostos. A maioria destes assume que a topologia da rede não é conhecida e, portanto, baseia-se em métodos probabilísticos para decidir se, em certo momento, uma certa mensagem deve ser transmitida ou não para um certo nó.
Se conhecemos as conexões a priori, podemos, utilizando Grafos Evolutivos, calcular jornadas de custo ótimo que não replicam nenhuma mensagem, diminuindo o tráfego da rede.

1. Redes Tolerantes a Atrasos

REDES Tolerantes a Atrasos (em inglês, Delay Tolerant Networks, ou DTNs) são redes que sofrem grandes mudanças de topologia e tem baixa conectividade. As principais características destas redes são:

- Atrasos variáveis
- Conexões intermitentes

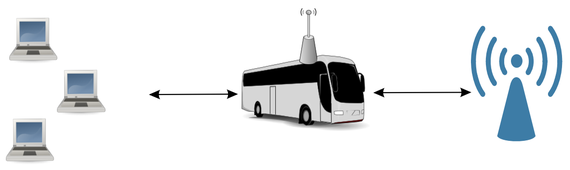
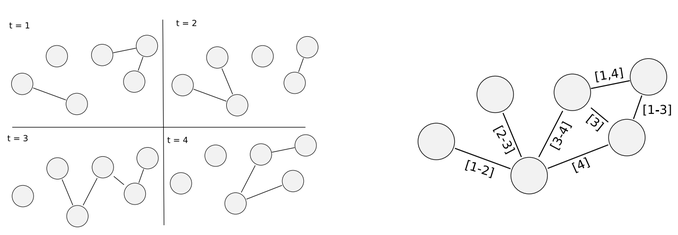


Figura 1: *Transmissão de dados em uma DTN*

Este tipo de rede é geralmente utilizado em regiões nas quais não há conexão direta com a Internet. Um veículo, conhecido como "mula" de dados, atravessa a região coletando dados gerados pelos usuários e os leva para o local mais próximo com conexão de banda larga, enviando os dados e obtendo as respostas.

2. Grafos Evolutivos

SE temos informações sobre todas as conexões que irão ocorrer na rede no decorrer do tempo, é possível construir um modelo para as DTNs que leve em conta tal conhecimento. Um modelo para estas redes chama-se *Grafo Evolutivo*.



(a) Conexões em quatro instantes de tempo.

(b) Grafo evolutivo correspondente a 2(a)

Figura 2: *Conexões e Grafo Evolutivo correspondente*

Definimos um grafo evolutivo da seguinte maneira:

Definição 1 (Grafos Evolutivos) *Sejam* $G = (V_G, E_G)$ *um grafo e* $S_G = G_0, G_1, ..., G_\tau$ *(*$\tau \in \mathbb{N}$*) um conjunto ordenado de subgrafos de* G *tal que* $\bigcup_{i=1}^{\tau} G_i = G$*. O sistema* $\mathcal{G} = (G, S_G)$ *é chamado de Grafo Evolutivo.*

Em um Grafo Evolutivo a noção de caminho deve ser alterada para representar o tempo de utilização de cada aresta (obviamente, uma aresta não pode ser percorrida em um tempo anterior ao tempo de transição da última aresta). Um caminho em que cada aresta tem um instante de tempo no qual deve ser percorrida é chamado de Jornada.

Definição 2 (Jornada) *Seja* $R = e_1, e_2, ..., e_k$ *(*$e_i \in E_G$*) um caminho em* G*, e* $R_\sigma = \sigma_1, \sigma_2, ..., \sigma_k$ *(*$\sigma_i \in [1, \tau]$*,* $\sigma_i \leq \sigma_j \ \forall i < j$*) um agendamento indicando quando cada arco de* R *deve ser atravessado. Então dizemos* $\mathcal{J} = (R, R_\sigma)$ *é uma jornada em* $\mathcal{G}$*.*

3. Jornadas Ótimas

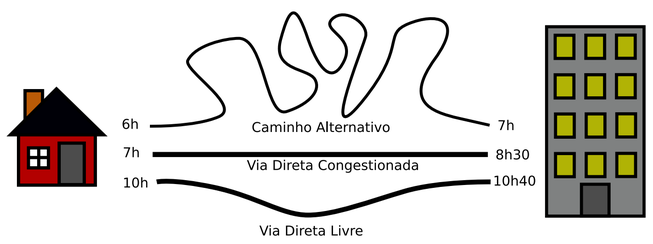


Figura 3: *Possíveis rotas de casa para o trabalho*

Podemos ver acima uma analogia da idéia de jornadas com as opções de percurso de uma pessoa indo de casa para o trabalho. Pode-se sair cedo e pegar um caminho alternativo (e comprido) para chegar o mais cedo possível no trabalho. Se ao invés do caminho alternativo, preferirmos seguir por uma via direta no horário de pico, o trânsito tornará o percurso mais demorado, apesar de tratar-se da rota mais curta. Se quisermos evitar o trânsito e passar o menor tempo possível na rua, podemos sair depois do horário de pico e pegar a via direta livre, chegando mais rápido.
Podemos, então, definir três tipos de jornadas ótimas em um grafo evolutivo, levando em conta três medidas distintas:

- O instante de chegada da mensagem ($\sigma_{|R|}$). Chamamos de *Foremost Journey* a jornada que chega no nó destino o mais cedo possível.
- O número de arcos percorridos ($|R|$). A jornada que percorre o menor número possível de arcos entre a origem e o destino é chamada *Shortest Journey*.
- O tempo de percurso, ou seja, a diferença entre o instante de envio (σ_1) e o instante de chegada ($\sigma_{|R|}$). A jornada com menor tempo de percurso é chamada *Fastest Journey*.

Algoritmo Foremost

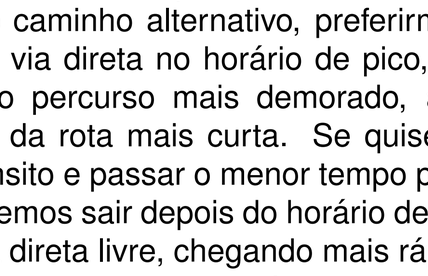


Figura 4: *Jornada Foremost*

Consumo de tempo: $O(M \log \delta_E + N \log N)$.

Algoritmo Shortest

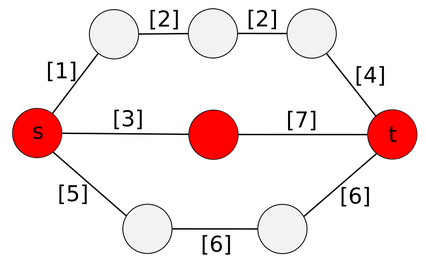


Figura 5: *Jornada Shortest*

Consumo de tempo: $O(N(M \log \delta_E + N))$

Algoritmo Fastest

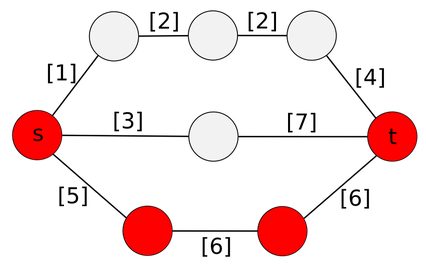


Figura 6: *Jornada Fastest*

Consumo de tempo $O(\mathcal{M}(Mlog\delta + N^2))$

4. Comparação

Comparando os três algoritmos acima, obtivemos os seguintes gráficos de tempo:

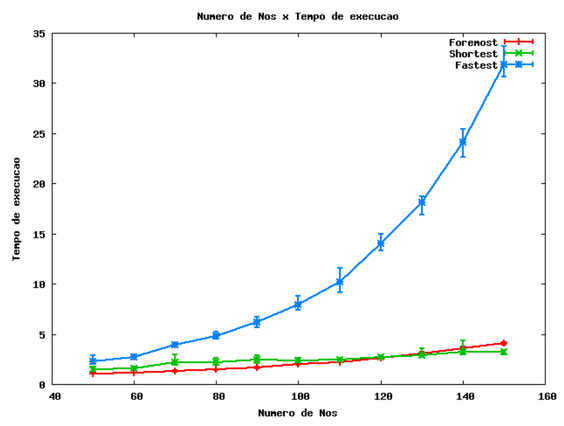


Figura 7: *Nós x Tempo de Execução (s)*

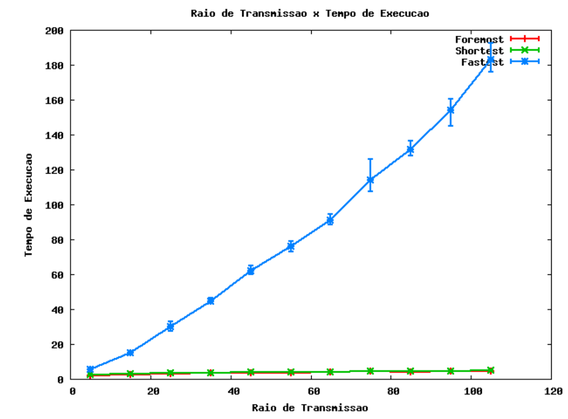


Figura 8: *Raio (m) x Tempo de Execução (s)*

Pelas figuras 7 e 8, podemos perceber que o algoritmo fastest é o mais demorado, o que é condizente com sua complexidade elevada. Já os algoritmos shortest e foremost não apresentam diferença significativa de tempo, apesar de a complexidade do shortest ser consideravelmente maior.

Referências

[1] Monteiro, J. and Goldman, A. and Ferreira, A., Using Evolving Graphs Foremost Journey to Evaluate Ad-Hoc Routing Protocols
[2] Xuan, B. and Ferreira, A. and Jarry, A., Computing shortest, fastest, and foremost journeys in dynamic networks

Este trabalho foi patrocinado pela CNPq