Internet packet routing: application of a K-quickest path algorithm

João Clímaco[1]

Marta Pascoal[1]

José Craveirinha[1]

M. Eugénia V. Captivo[2]

jclimaco@inescc.pt, marta@mat.uc.pt, jcrav@dee.uc.pt, mecaptivo@fc.ul.pt

[1]University of Coimbra [2]University of Lisbon

PORTUGAL

Abstract

This paper describes a study on the application of an algorithm to rank the K quickest paths to the routing of data packets in Internet networks.

For this purpose an experimental framework was developed by considering two types of random generated networks.

To obtain realistic values of the IP packet sizes, a truncated Pareto distribution was defined, having in mind to reflect a key feature of Internet traffic, namely its self-similar stochastic nature.

Results concerning the average CPU times of the algorithm for the different sets of experiments will be presented and discussed.

Contents

1. Internet packet routing
2. K-quickest path algorithm
   (a) Problem and generic deviation algorithm
   (b) Set of paths partition and quickest paths computation
   (c) Quickest path deviating from another before a given node
3. Application model: Simulation of Internet packet routing
   (a) Packet sizes σ: (truncated) Pareto distribution
   (b) Arc delays and bandwidths
4. Application model: Experiments set
5. Computational results
6. Conclusions

Internet packet routing

Traditionally routing of packets in the Internet is based on shortest path algorithms using additive metrics (as count or delay).
However, more effective routing approaches include **path multiple metrics** (as bandwidth, reliability, link load).

Cisco's routing protocol EIGRP uses an objective function that represents the time to transmit σ data units throughout path p:

$$T(p) = d(p) + \frac{\sigma}{B(p)} \, .$$

- Delay of p: $d(p) = \sum_p d_{ij}$
- Lowest bandwith of p: $B(p) = \min_p \{b_{ij}\}$

Quickest path problem

Internet packet routing

Quickest path problem was solved by:

- Rosen *et al.* (1991)
- Martins & Santos (Bricriteria problem $\min\{\sum d_{ij}\}$, $\max\{\min\{b_{ij}\}\}$) (1997)

Algorithm:

- Compute shortest paths (for delay) with increasing minimum bandwidths
- Choose the one with minimum transmission time

Worst-case time complexity	$\mathcal{O}(r(m + n \log n))$
Worst-case space complexity	$\mathcal{O}(m + n)$

In a network with n nodes, m arcs, and r distinct bandwidth values.

K-quickest path algorithm

Given $K > 1$, we intend to compute routes

$$p_1, \ldots, p_K$$

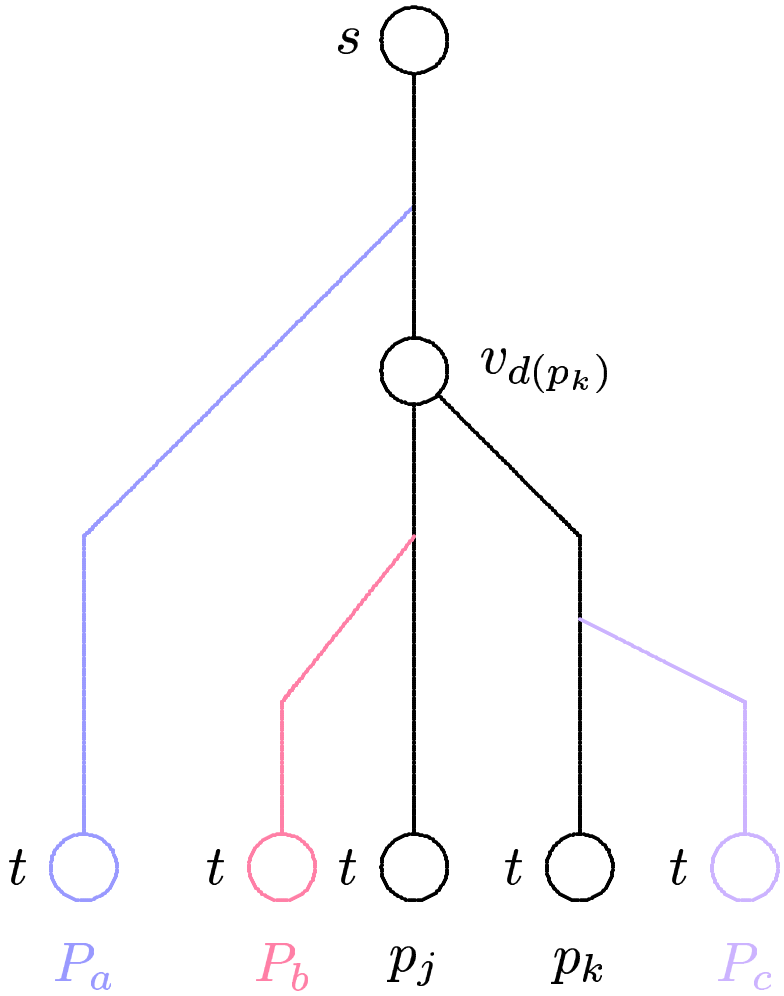
from s to t in an **undirected network** $(\mathcal{N}, \mathcal{A})$, by **non-decreasing order of** T.

Deviation algorithm that uses a set X of candidates to k-th quickest path, $k \in \{1, \ldots, K\}$.

Algorithm:

- Compute the quickest path; Store it in X
- For ($k \in \{1, \ldots, K\}$) Do
  - $p_k \leftarrow$ best element in X
  - $X \leftarrow X - \{p_k\}$
  - Analyse p_k and compute new candidates; Store them in X

K-quickest path algorithm



New quickest paths:

P_a: deviating from p_j before $v_{d(p_k)}$

P_b: deviating from p_j after or at $v_{d(p_k)}$

P_c: deviating from p_k after $v_{d(p_k)}$

K-quickest path algorithm

Let $(\mathcal{N}, \mathcal{A}(b_i))$ be the network obtained from $(\mathcal{N}, \mathcal{A})$ ignoring arcs with bandwidth lower than b_i.

Let v_α be a node of path

$$p = \langle s = v_1, v_2, \ldots, t = v_{\ell(p)} \rangle$$

The quickest path deviating from p before v_α is:

- A shortest path q that separates from p before v_α in some of the networks $(\mathcal{N}, \mathcal{A}(b_i))$, $i = 1, \ldots, r$.

K-quickest path algorithm

Let:

- $\mathcal{T}_s$ ($\mathcal{T}_t$) be the tree of shortest paths from s to any node (from any node to t)
- $\mathcal{T}_s(i)$ ($\mathcal{T}_t(i)$) be the path from s to i in $\mathcal{T}_s$ (from i to t in $\mathcal{T}_t$)
- $\xi_s(i)$ be the index of the node where $\mathcal{T}_s(i)$ deviates from $p^* = \mathcal{T}_s(t) = \mathcal{T}_t(s)$ (analogously for $\mathcal{T}_t(i)$)

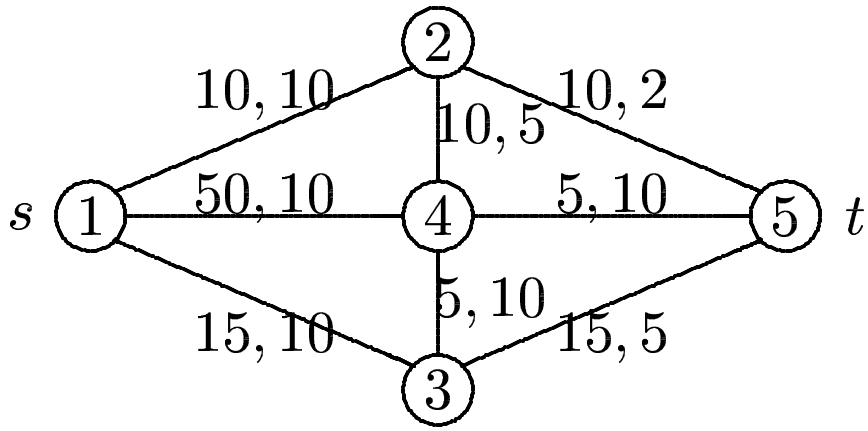
Path q can be found according to the following scheme:

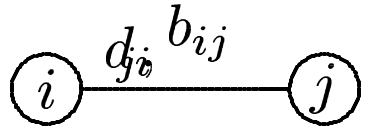
- If $p \neq \mathcal{T}_s(t)$ then $q = \mathcal{T}_s(t) = \mathcal{T}_t(s)$
- Else q is the minimum delay path of type 1 or type 2, where:

  Type 1. $\mathcal{T}_s(i) \diamond \mathcal{T}_t(i)$, with $\xi_s(i) < \alpha$

  Type 2. $\mathcal{T}_s(i) \diamond (i,j) \diamond \mathcal{T}_t(j)$, with $(i,j) \in \mathcal{A} - (\mathcal{T}_s \cup \mathcal{T}_t)$ and $\xi_s(i) < \alpha$

Example



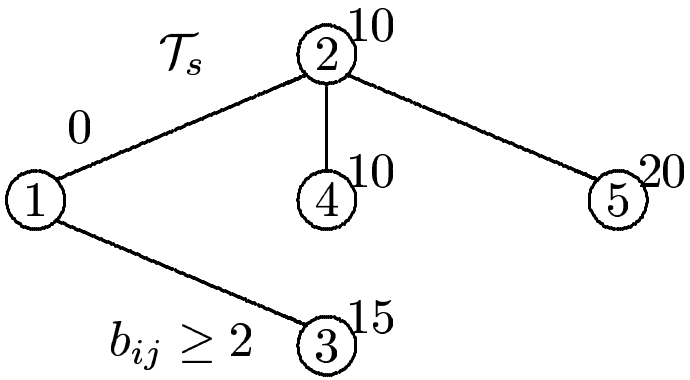


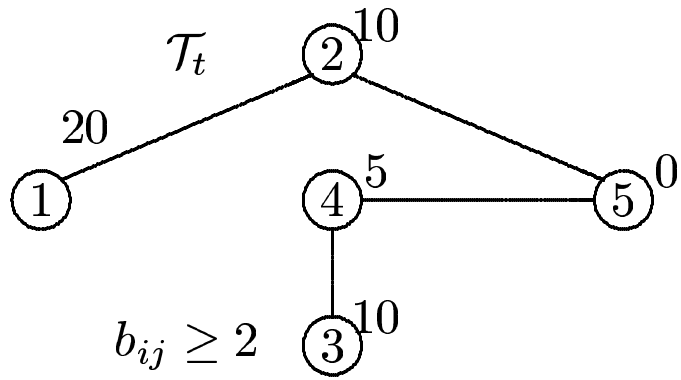
$\sigma = 100$

$$\left.\begin{array}{lll} (\mathcal{N}, \mathcal{A}(2)): & p = \langle 1,2,5 \rangle & T(p) = 70 \\ (\mathcal{N}, \mathcal{A}(5)): & p = \langle 1,2,4,5 \rangle & T(p) = 45 \\ (\mathcal{N}, \mathcal{A}(10)): & p = \langle 1,3,4,5 \rangle & T(p) = 35 \end{array}\right\} \Rightarrow \quad p_1 = \langle 1,3,4,5 \rangle$$

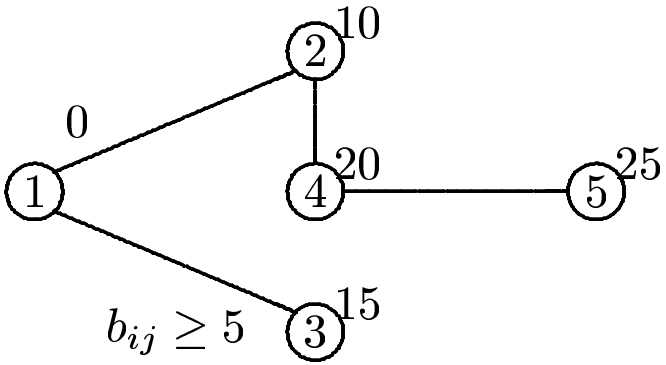
Compute the quickest path that deviates from p_1 before 5.

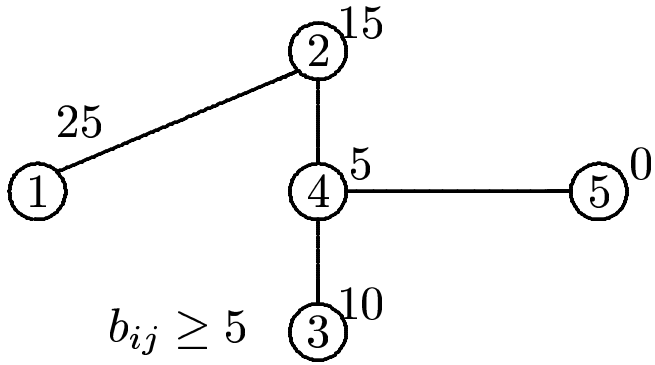
Example



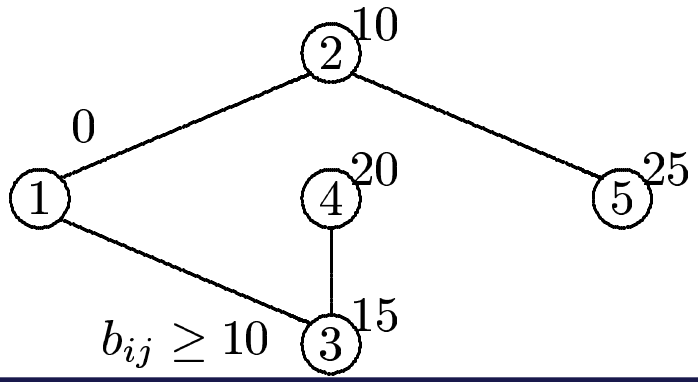


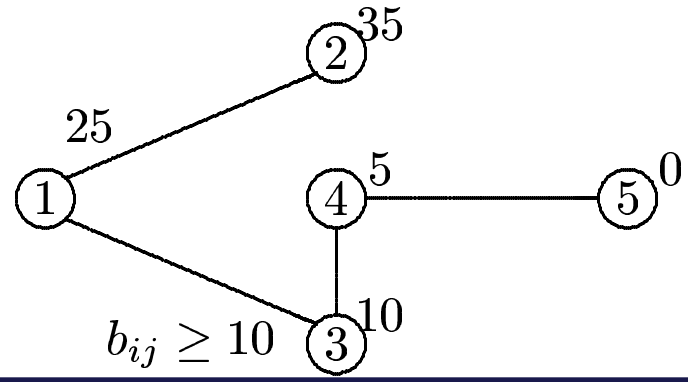
$p \neq \mathcal{T}_s(t)$
$q = \langle 1, 2, 5 \rangle$
$T(q) = 70$





$p \neq \mathcal{T}_s(t)$
$q = \langle 1, 2, 4, 5 \rangle = p_2$
$T(q) = 45$

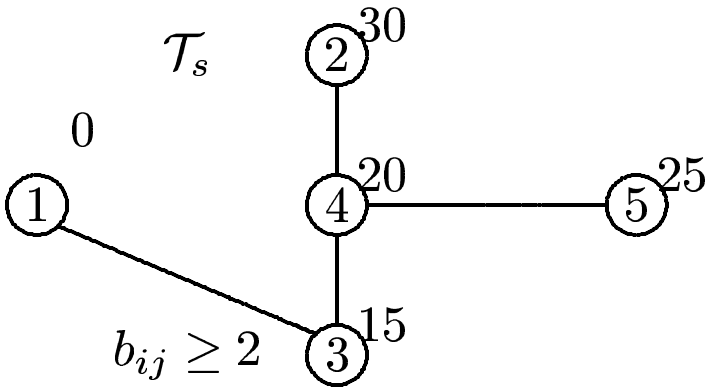


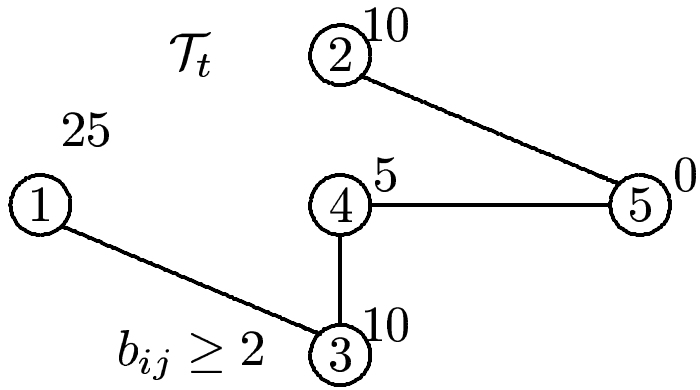


$p = \mathcal{T}_s(t)$
$q = \langle 1, 4, 5 \rangle$
$T(q) = 65$

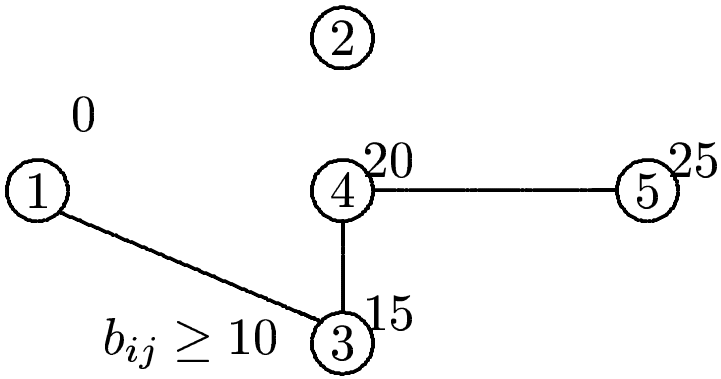
Example

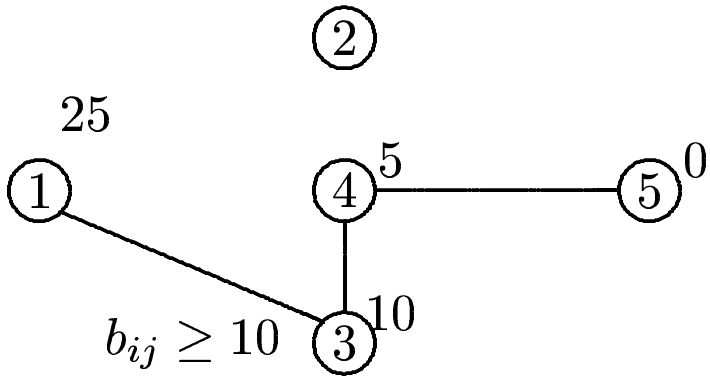
Remove node 1, compute the quickest path deviating from $\langle 2, 4, 5 \rangle$ before 5.





$p = \mathcal{T}_s(t)$
$q = \langle 1, 3, 5 \rangle = P_b$
$T(q) = 50$

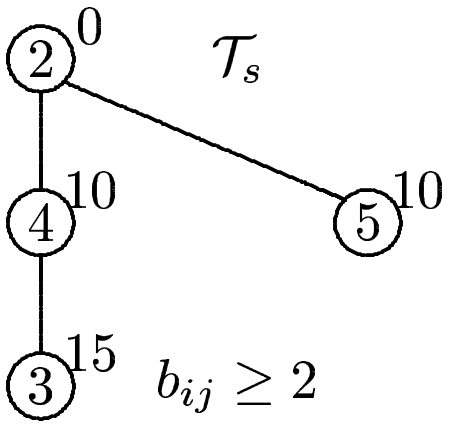


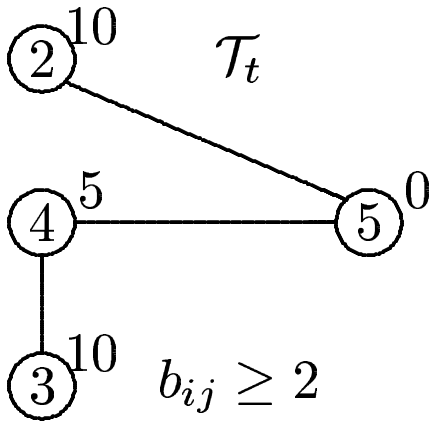


$p = \mathcal{T}_s(t)$
$q = \langle 1, 4, 5 \rangle$
$T(q) = 65$

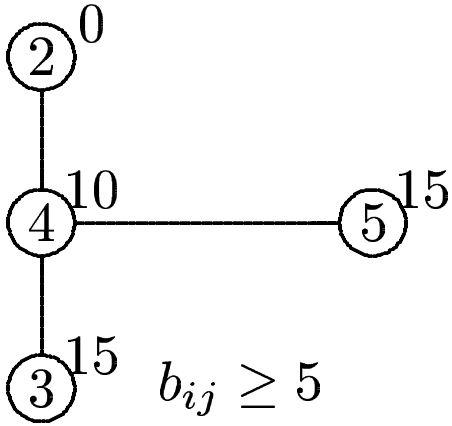
Example

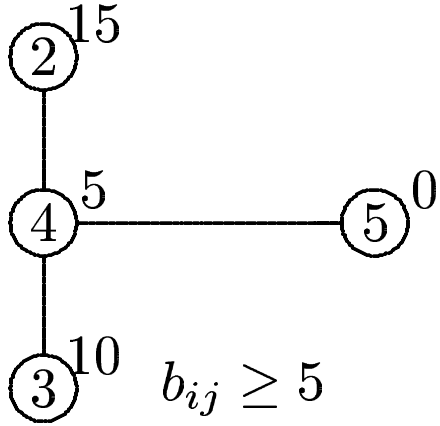
Remove arc $(1,2)$, compute the quickest path deviating from p_1 before 5.





$p \neq \mathcal{T}_s(t)$
$q = \langle 1, 2, 5 \rangle$
$T(q) = 70$





$p = \mathcal{T}_s(t)$
$q = \langle 1, 2, 4, 3, 5 \rangle = P_a$
$T(q) = 60$

$$X = \{\langle 1, 3, 5 \rangle, \langle 1, 2, 4, 3, 5 \rangle\} \Rightarrow p_3 = \langle 1, 3, 5 \rangle$$

Simulation of Internet packet routing

Internet traffic has a **self-similar property** (identical key statistical features in all time scales).

$\Downarrow$

The correlation function of a discrete stationary self-similar traffic process decays asymptotically according to a hyperbolic function.

$\Downarrow$

Heavy tailed distributions

A practical distribution often used in this context is the **Pareto distribution**:

$$F_X(x) = P(X \leq x) = 1 - \left(\frac{b}{x}\right)^{\alpha},\ x \in [b, +\infty[,\ \alpha, b > 0,$$

- b is the location parameter,
- $\alpha \in]1, 2[$ guarantees the statistical properties required by traffic processes with long range dependence.

Simulation of Internet packet routing

Maximum IP datagram format size (header and content): $M = 524280$ bits.
Minimum size (Ethernet II transmission unit): $b = 512$ bits.

Packet sizes $\sigma \in [b, M]$ simulated with a **truncated Pareto distribution**:

$$\hat{F}_\sigma(x) = P(\sigma \leq x) = \frac{1 - (b/x)^\alpha}{F_\sigma(M)}$$

To compute σ values, generate uniformly $u \in [0, 1]$, and let

$$x = \hat{F}^{-1}(u) \Longrightarrow x = b\left(1 - uF_\sigma(M)\right)^{-1/\alpha}$$

$H = \dfrac{3-\alpha}{2}$ (Hurst parameter) measures self-similarity described by the number of busy servers of the M/G/∞ queue with Pareto distributed service times. We used:

$$\alpha = 1.2\ (H = 0.9) \ \text{ and } \ \alpha = 1.5\ (H = 0.75)$$

Simulation of Internet packet routing

Delays and bandwidths obtained from results in a study on packet dynamics.

d_{ij}	11	16	25	42	73	128	227	410	744	1365	2520	4681	8700
%	2.3	5.4	8.5	10	12	11	10	11	8.5	7	5	5	4.3

Arcs delay distribution (in ms)

b_{ij}	1360	64	128	256	800	1680	2640	4000	8000
%	51.30	7.15	5.30	0.88	4.40	19.47	4.40	2.70	4.40

Arcs bandwidth distribution (in bits/ms)

Experiments set

Undirected networks (connected) with n nodes and $m = 4n$ arcs.

Each node is adjacent to at least 2 and at most 10 other nodes.

Two classes of networks:

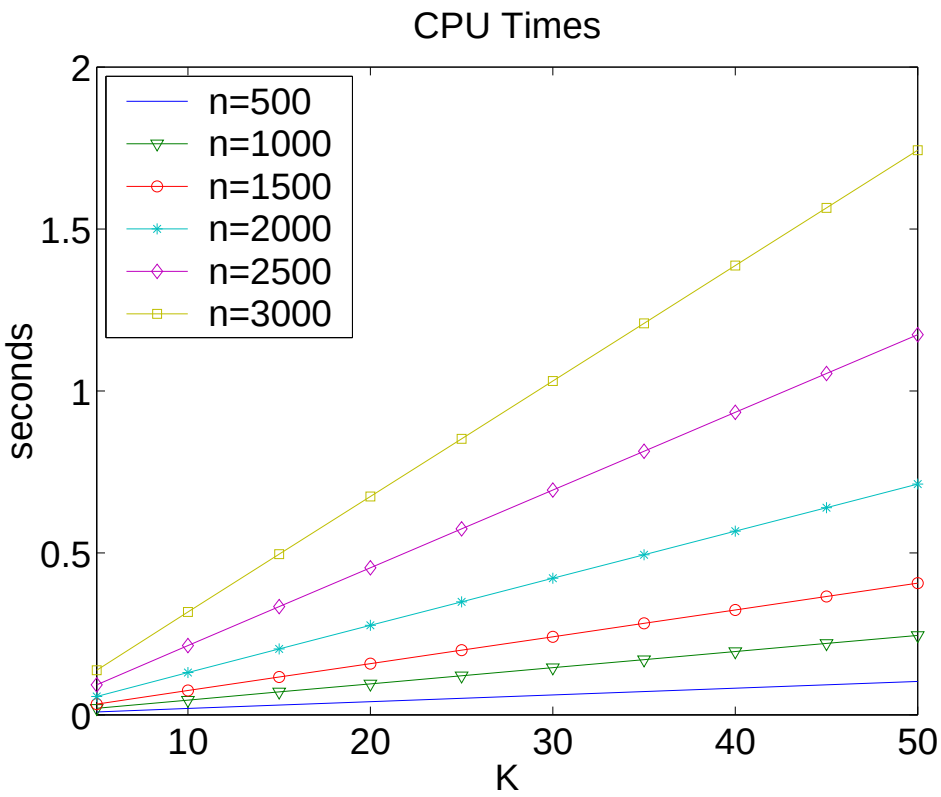
- Based on a rectangular 400×240 grid, where $n \in \{500, 1000, \ldots, 3000\}$ nodes are randomly chosen
- Based on geographic coordinates of $n = 1088$ US cities (`http://www.realestate3d.com/gps/latlong.htm`)

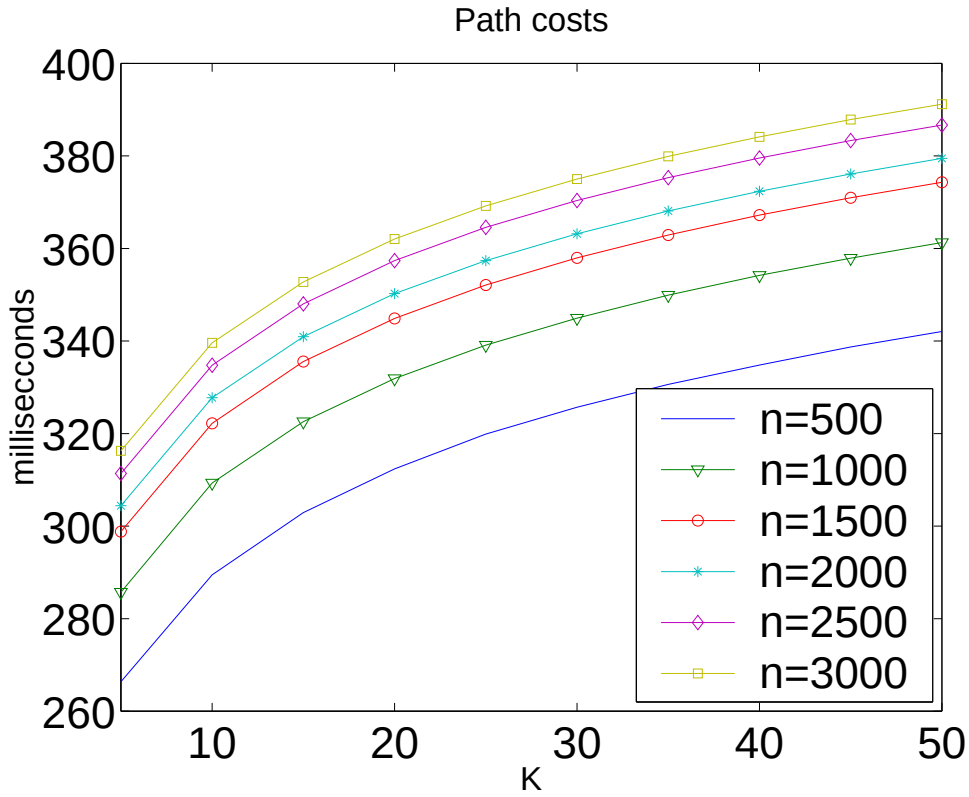
generated with 10 different seeds and $\frac{n^2}{2500}$ s-t node pairs.

Code and computer:

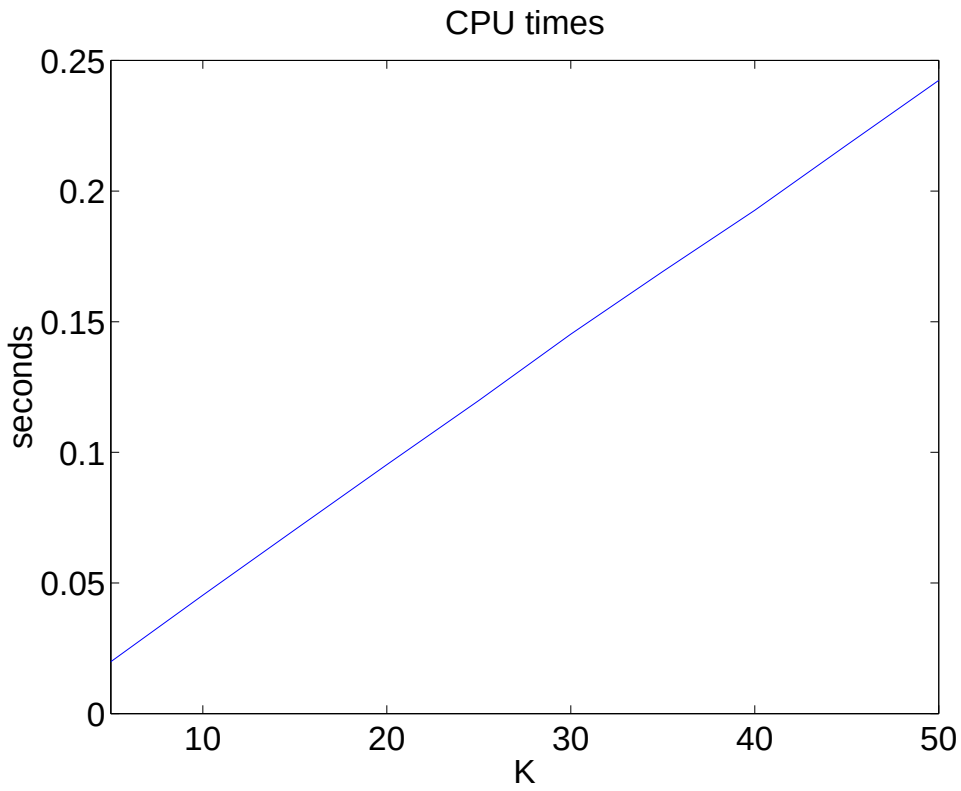
- C language on Linux,
- AMD Athlon workstation at 1.3 GHz, with 512 Mb of RAM.

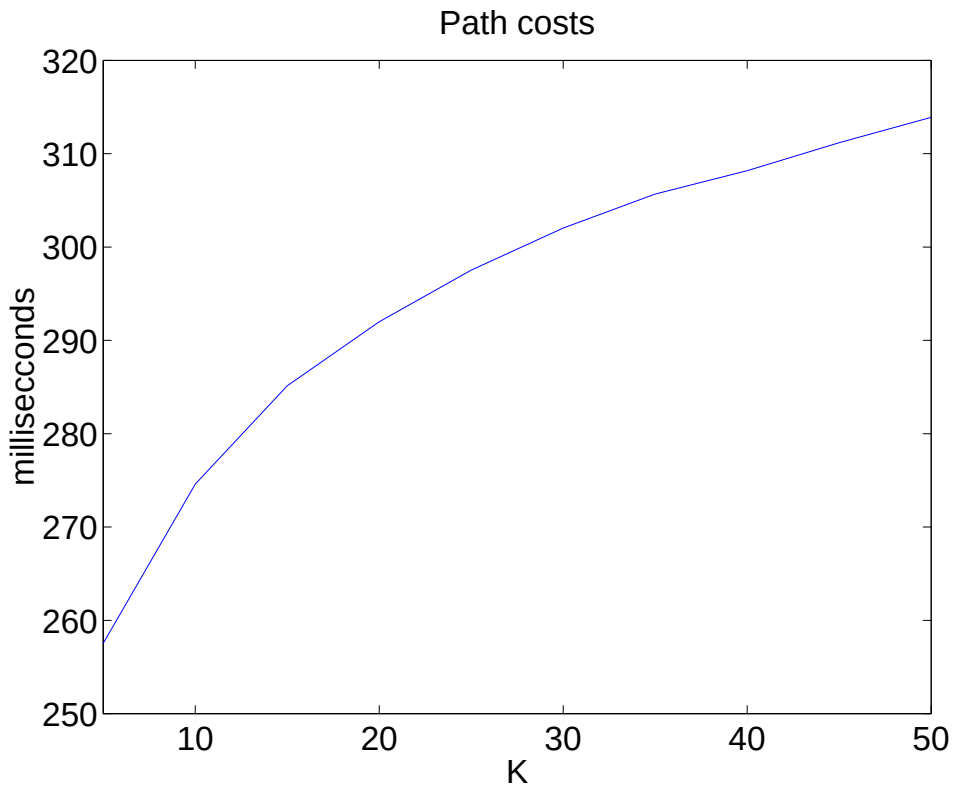
Results – Random networks





Results – US network





Conclusions

Proposals:

- Generation of alternative routes for packets in Internet, using total trasmission time as the metric function and applying an efficient K-quickest path algorithm.
- Probabilistic model to generate packet sizes, using a truncated Pareto distribution to simulate realistic Internet routing conditions.
- Computational experience on 2 topologies: random networks and US-cities networks.

Empirical results:

- CPU times depend linearly on the number of determined paths and increase with the size of the network.
- The procedure is efficient in practical situations.
- In the worst tested situation the 50 best solutions in a network with 3000 nodes were obtained in 1.743 seconds.

References

Chen. An algorithm for finding the K quickest paths in a network. *Comp. & Op. Res.*, 20:59–65, 1993.

Martins & Santos. An algorithm for the quickest path problem. *Op. Res. Letters*, 20:195–198, 1997.

Park & Willinger, editors. *Self-similar network traffic and performance evaluation*, ch 1. Self-similar network traffic: An overview, pp 1–38. Wiley-Interscience, 2000.

Pascoal, Captivo, and Clímaco. An algorithm for ranking quickest simple paths. To appear in *Comp. & Op. Res.*, 2003. (`http://www.mat.uc.pt/~marta/Publicacoes/rank_quick.ps.gz`).

Paxson. End-to-end Internet packet dynamics. *IEEE/ACM Transactions on Networking*, 7:277–292, 1999.

Riedl & Schupke. A flow-based approach for ip traffic engineering utilizing routing protocols with multiple metric types. In *Proc. of the 6th INFORMS Telecom. Conf.*. Boca Raton, USA, 2002.

Rosen, Sun, and Xue. Algorithms for the quickest path problem and the enumeration of quickest paths. *Comp. & Op. Res.*, 18:571–584, 1991.

Savage, Collins, Hoffman, Snell, and Anderson. The end-to-end effects of Internet path selection. In *Proc. of ACM SIGCOMM'99*, pp 289—299, 1999.

Wilder. *A Guide to the TCP/IP Protocol Suite*. Artech House, Inc., Boston, 1998.