Quantum Shortest Path Netsukuku

http://netsukuku.freaknet.org
AlpT (@freaknet.org)

December 13, 2006

Abstract

This document describes the QSPN, the routing discovery algorithm used by Netsukuku. Through a deductive analysis the main proprieties of the QSPN are shown. Moreover, a second version of the algorithm, is presented.

1 Preface

1. Mobiles node aren't supported by Netsukuku algorithms. [1]
2.

For the sake of simplicity, in this paper, we will assume to operate on level 0 (the level formed by 256 single nodes).

4 Tracer Packet

A *TP* (Tracer Packet) is the fundamental concept on which the QSPN is based: it is a packet which stores in its body the IDs of the traversed hops.

4.1 Tracer Packet flood

A TP isn't sent to a specific destination but instead, it is used to flood the network. By saying "the node A sends a TP" we mean that "the node A is starting a TP flood".

A TP flood passes only once through each node of the net: a node which receives a TP will forward it to all its neighbours, except the one from which it

At the end, A will know the route $A \to B \to C \to D$ and F will know the route $F \to E \to D$.

```
            current branch can't be explored anymore, therefore it is a
            valid route. Print it */
        print branch
}
```

A proof of concept of the above algorithm has been implemented in Awk [4].

Example

Consider this graph:

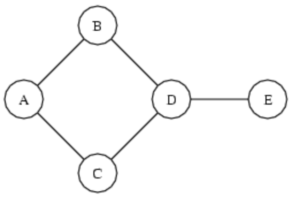


Figure 2: A simple graph with one segment and one cycle

Given this graph as input the algorithm will output:

$$
\begin{array}{l}
A \to B \to D \to C \\
A \to B \to D \to E \\
A \to C \to D \to B \\
A \to C \to D \to E \\
B \to A \to C \to D \to E \\
B \to D \to C \to A \\
B \to D \to E \\
C \to A \to B \to D \to E \\
C \to D \to B \to A \\
C \to D \to E \\
D \to B \to A \to C \\
D \to C \to A \to B \\
D \to E
\end{array}
$$

6 Raw Tracer Packet flood

We can consider each route given by the output of the above algorithm as a

singT310(341-o)-31(o)t40-334(g42(s)-1ot4(g42(r)-1(r)-3st(thm)(t,)-4525()41-like)-420()41-in0()41-a0()41-n(it node doesn’t forward the RTP to the neighbour from which it has received the packet itself.

Xc. . . c + XcY   Xc. . . cY Example:

$$123ABCDA + 123A987 \quad 123ABCDA987$$

c. . . cZ + YcZ   Yc. . . cZ Example:

$$ABCDA123 + 987A123 \quad 987ABCDA123$$

c. . . c + YcZ   Yc. . . cZ Example:

$$ABCDA + 987A123 \quad 987ABCDA123$$

Invalid route A route must not be in the form of:

$$XacaY$$

where *a* and *c* are two nodes. A simplification, which gives a route of this

3. In a cycle, just two TP are needed, and one is the reverse of the other. The first can be constructed in this way:

   - Choose a node of the cycle, this will be the pivot node.
   - Start from one neighbour of the pivot and write sequencially all the other nodes until you return to the pivot (but do not include it). Call this string C.
   - The TP will be:

     $$CpC$$

     where p is the pivot node.

   Example: if we choose the node D as the pivot, we can write the TP as:

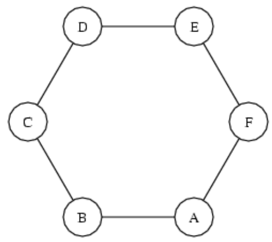


Figure 4: A cycle

$$EFABCDEFABC$$

and its reverse:

$$CBAFEDCBAFE$$

These two TPs will give all the routes to all the nodes of the cycle.

7.3 The question

Can we implement a “live” version of the Simplify Route algorithm like we did with the Generate Route one?
The reply is ahead.

8 Continuous Tracer Packet

A Continuous Trker Plket (CTP) is an extension of the TP flood: a node will

9.1 Interesting information

A node considers a received CTP interesting when its body contains at least a new route, i.e. a route that the node didn't previously know. In other words, if a CTP contains routes already known by the node, it is considered uninteresting.

9.3 Cyclicity

rtt or the bandwidth capacity. If the node has reached the *MaxRoutes* limit, it will substitute the old route with the more efficient one.

Note that this definition is more general than the previous. Indeed, if the node S doesn't know the route to reach D

The underlined routes are the new route for *G*. As you can see, in the CTP (10) G doesn't find any new route, so it drops the packet and doesn't

5. Q^2 is easier and simpler than Q to be implemented. In general this means

because the old routes can contain interesting information about upload routes. For example, consider this segment:

$$\cdots \leftrightarrow A \leftrightarrow B \leftrightarrow C \leftrightarrow N$$

If N doesn't erase the route received in the CTP, A will receive the following CTP:

$$\cdots \rightarrow A \rightarrow B \rightarrow C \rightarrow N \rightarrow C \rightarrow B \rightarrow A$$

In this case A will know the following upload route:

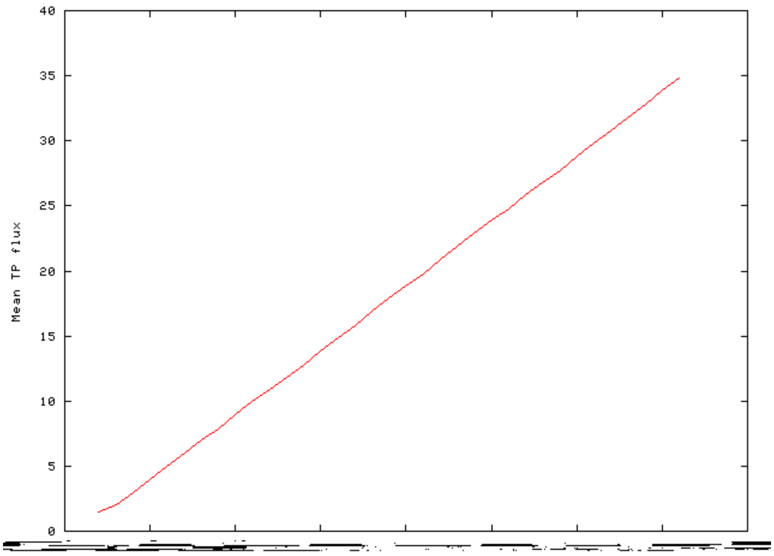
$$A \rightarrow B \rightarrow C \rightarrow N$$

When parsing a CTP, a node will recognize the part of the routes which are in the form of $XacaY$, where a, c are two nodes and X and Y are two generic routes. The packet will then be split in Xac and caY.

At this point we've finished. In fact, we are sure to receive at least one upload route per node because a CTP traverses each path first in one sense and then

2. for each memorised route S we compute $s(R, S)$ and if we find a route S such that $s(R, S) > 0.5$ we go to step 3.

3. we set
$$R_e = R_e{}^{1 - s(R, S)}$$


40
35
30
25
20
15
10
5
0
Mean TP flux

where n is the number of nodes of the graph. For example, in random graphs with increasing number of nodes, the mean TP flux will assume the distribution shown in the figure below:

The sa3wna3wfurd[(Thead[(The)-282()-354hd[(T3m)-36r)1(aph)0792d[(The)-282()-354hd[(T3m)-36rT3mu

References

[1] Netsukuku website: http://netsuk4(h)28983sr98eakn98e1983org/

[2] Netsukuku topology document: topology.pdf1(u)1(kuk)1(u)-333(top)-2e-goA.wu66427(df9rlogy)-324.80

^_^