Supplementary Information for

Multipath routing based on a network coding protocol suit

1. The Setting of Numerical Simulations

In the simulations, the probability that there is an edge between two nodes is set to 0.5. Given the number of nodes, the network topology is randomly generated for 10000 times, each time two nodes being randomly selected as the source node and the destination, respectively. The length of the binary strings is set to 8, which corresponds to the size of finite field $F = GF(2^8)$.

For each generated network topology, the same search algorithm is executed by each node to generate a multipath routing table to support NC. Specifically, after the complete multi-path routing table being built at node v, it is convenient for v to know which edge can act as the incoming link from the source node and which one can act as the outgoing (forward) link to the destination, which means the size of local coding matrix is determined.

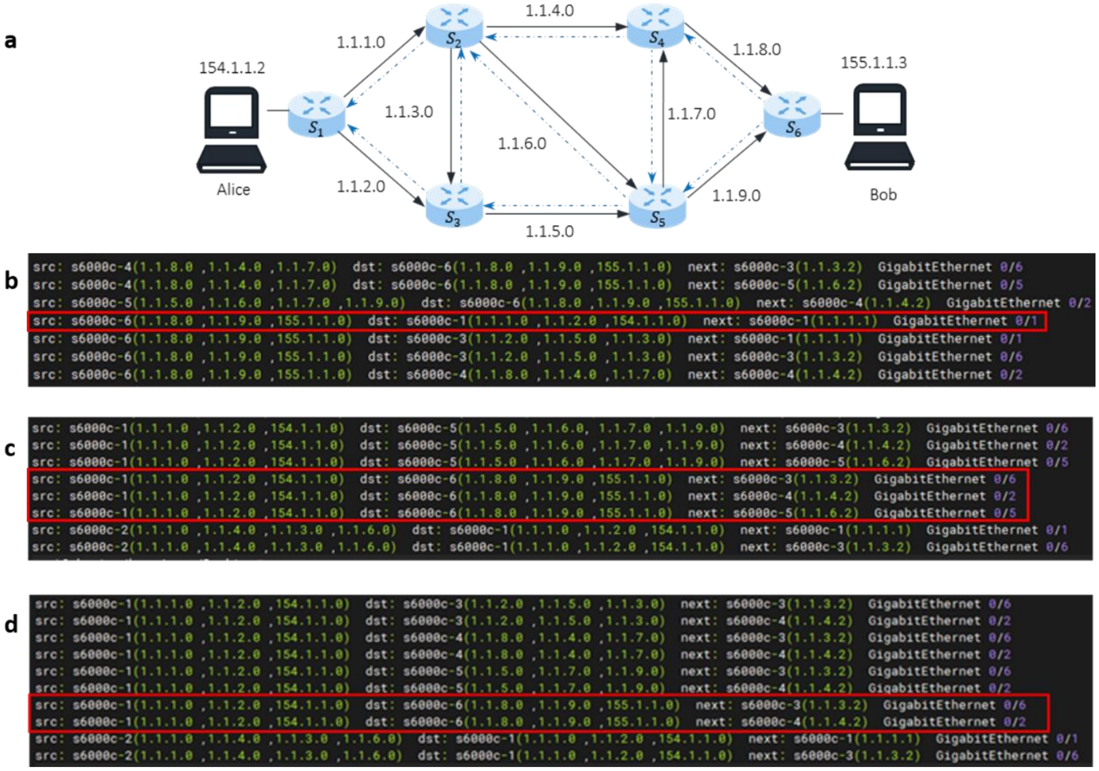
As shown in Fig. 2a, it is easy to form a DA graph between the source router and the destination by connecting each selected node with the next hops to the destination, given the multi-path routing tables for all selected nodes.

2. An Engineering Verification

An implementation of the proposed protocol suit is shown in Supplementary Fig. 1a, where S_1, S_2, …, S_6 are NC-integrated layer-3 switches (the model is RG-S6000C, produced by ************ Communication Company), the dotted lines represent the reverse links while the solid lines are the forward links of the generated DA graph between the source S_1 and the destination S_6, and the IP addresses near the lines indicate the different network segments the links belong to. Alice accesses the network via switch S_1 and Bob accesses the network via S_6. Alice's IP address is set to 154.1.1.2 and Bob's address is 155.1.1.3. All the layer-3 switches are developed on the same Linux kernel, each having 6.16T switching bandwidth, 252M packet-forwarding rate, and 48 gigabit Ethernet ports. In this verification, some functions of the intermediate system-to-intermediate system (IS-IS) protocol are applied to obtain the adjacent matrix of the whole network, and all network segments the links belong to can also be obtained (Supplementary Fig. 1b-d). The protocol number of NC-related packets is temporarily set to 234 to distinguish them from other packets.

Supplementary Fig. 1b-d show the generated routing entries at switch S_2 under different circumstances, where the string src refers to the source switch, dst represents the destination, $next$ refers to the next hop, and $s6000c$ is the model of the layer-3 switches. As shown in Supplementary Fig. 1b, switch S_2 has only one next hop (S_1 exactly) to the source switch S_1 along the reverse link. On contrast, there are three next hops (S_3, S_4, and S_5) to the destination S_6 along the forward links, which is shown in Supplementary Fig. 1c. Once the cable connecting switch S_2 and S_5 has been removed, switch S_5 will no longer serve as the next hop to the destination, which is shown in Supplementary Fig. 1d. This verifies that dynamic routing is well supported in the proposed protocol suit.

An example of the encoding matrices generated for the switches is shown in Supplementary Fig. 2, where the word length of matrix elements is 2 bytes and the data is represented in decimal system for convenience.

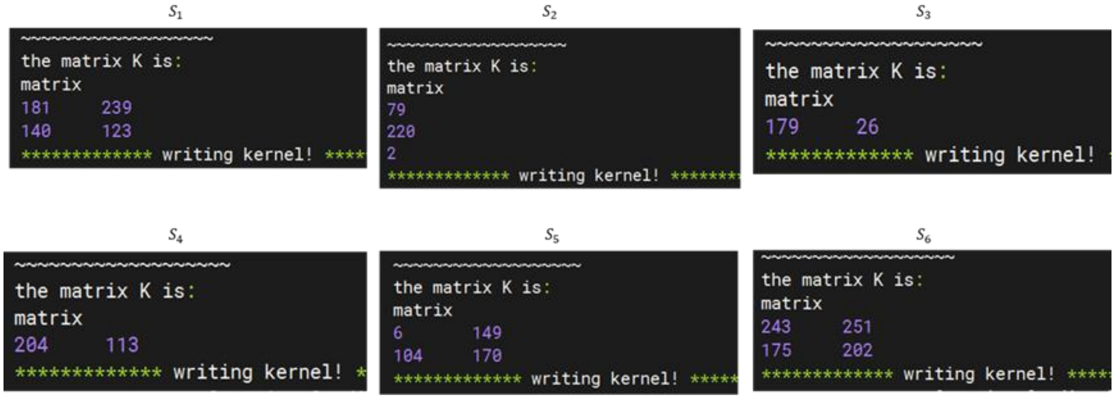


```
src: s6000c-4(1.1.8.0 ,1.1.4.0 ,1.1.7.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-4(1.1.8.0 ,1.1.4.0 ,1.1.7.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-5(1.1.6.2)  GigabitEthernet 0/5
src: s6000c-5(1.1.5.0 ,1.1.6.0 ,1.1.7.0 ,1.1.9.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
src: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  dst: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  next: s6000c-1(1.1.1.1)  GigabitEthernet 0/1
src: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  dst: s6000c-3(1.1.2.0 ,1.1.5.0 ,1.1.3.0)  next: s6000c-1(1.1.1.1)  GigabitEthernet 0/1
src: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  dst: s6000c-3(1.1.2.0 ,1.1.5.0 ,1.1.3.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  dst: s6000c-4(1.1.8.0 ,1.1.4.0 ,1.1.7.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
```

```
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-5(1.1.5.0 ,1.1.6.0, 1.1.7.0 ,1.1.9.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-5(1.1.5.0 ,1.1.6.0 ,1.1.7.0 ,1.1.9.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-5(1.1.5.0 ,1.1.6.0 ,1.1.7.0 ,1.1.9.0)  next: s6000c-5(1.1.6.2)  GigabitEthernet 0/5
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-5(1.1.6.2)  GigabitEthernet 0/5
src: s6000c-2(1.1.1.0 ,1.1.4.0 ,1.1.3.0 ,1.1.6.0)  dst: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  next: s6000c-1(1.1.1.1)  GigabitEthernet 0/1
src: s6000c-2(1.1.1.0 ,1.1.4.0 ,1.1.3.0 ,1.1.6.0)  dst: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
```

```
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-3(1.1.2.0 ,1.1.5.0 ,1.1.3.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-3(1.1.2.0 ,1.1.5.0 ,1.1.3.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-4(1.1.8.0 ,1.1.4.0 ,1.1.7.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-4(1.1.8.0 ,1.1.4.0 ,1.1.7.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-5(1.1.5.0 ,1.1.7.0 ,1.1.9.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-5(1.1.5.0 ,1.1.7.0 ,1.1.9.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
src: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  dst: s6000c-6(1.1.8.0 ,1.1.9.0 ,155.1.1.0)  next: s6000c-4(1.1.4.2)  GigabitEthernet 0/2
src: s6000c-2(1.1.1.0 ,1.1.4.0 ,1.1.3.0 ,1.1.6.0)  dst: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  next: s6000c-1(1.1.1.1)  GigabitEthernet 0/1
src: s6000c-2(1.1.1.0 ,1.1.4.0 ,1.1.3.0 ,1.1.6.0)  dst: s6000c-1(1.1.1.0 ,1.1.2.0 ,154.1.1.0)  next: s6000c-3(1.1.3.2)  GigabitEthernet 0/6
```

SUPPLEMENTARY FIG. 1. **An implementation of the multipath search algorithm. a,** A real transmission network with 6 NC-integrated layer-3 switches. All the nodes and the solid directed lines connecting them consist of a directed acyclic graph exactly between router $\boldsymbol{S_1}$ and $\boldsymbol{S_6}$. (**b** to **d**) show the routing entries generated at node $\boldsymbol{S_2}$ under different circumstances. **b,** Only one next hop to the router $\boldsymbol{S_1}$. **c,** Three next hops to the router $\boldsymbol{S_6}$. **d,** Two next hops remain to the destination. Dynamic routing is well supported in the algorithm.



S_1

```
~~~~~~~~~~~~~~~~~~~~
the matrix K is:
matrix
181     239
140     123
************* writing kernel! ****
```

S_2

```
~~~~~~~~~~~~~~~~~~~~
the matrix K is:
matrix
79
220
2
************* writing kernel! *******
```

S_3

```
~~~~~~~~~~~~~~~~~~~~
the matrix K is:
matrix
179     26
************* writing kernel!
```

S_4

```
~~~~~~~~~~~~~~~~~~~~
the matrix K is:
matrix
204     113
************* writing kernel! *
```

S_5

```
~~~~~~~~~~~~~~~~~~~~
the matrix K is:
matrix
6       149
104     170
************* writing kernel! *****
```

S_6

```
~~~~~~~~~~~~~~~~~~~~
the matrix K is:
matrix
243     251
175     202
************* writing kernel! *****
```

SUPPLEMENTARY FIG. 2. **An example of the generated encoding matrices at 6 switches.**