Routing algorithm taxonomies

We characterize routing along two axes, representing how and when routing decisions are made

How:

Deterministic => known source and destination nodes have a single path taken between them that is completely defined. No path diversity with deterministic routing.

Non-deterministic => path diversity exists, and paths between source and destination are chosen dynamically.

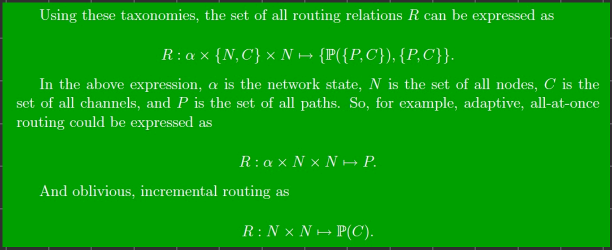
Oblivious => paths are randomly selected

Adaptive => additional state information is used to inform routing decision. This information can include local channel loads, past path choices, buffer availability, etc. Global information is generally not useful because of the race condition in assembling this information vs. the staleness of the information.

When:

All-at-once (source-routing): path of a message is determined completely at the source node. This reduces the number of routing decisions to one per packet, but prevents the use of state at other nodes of the network (presumably planning ahead for potential collision avoidance). This is useful if and when there’s not enough time to perform a routing decision at every hop.

Incremental: only the next hop is determined at a particular node. Routing decision per packet equals number of hops and state of each intermediate node can be used in the routing decision.



Using these taxonomies, the set of all routing relations R can be expressed as

$$R : \alpha \times \{N, C\} \times N \mapsto \{\mathbb{P}(\{P, C\}), \{P, C\}\}.$$

In the above expression, α is the network state, N is the set of all nodes, C is the set of all channels, and P is the set of all paths. So, for example, adaptive, all-at-once routing could be expressed as

$$R : \alpha \times N \times N \mapsto P.$$

And oblivious, incremental routing as

$$R : N \times N \mapsto \mathbb{P}(C).$$

Table-based Routing

Routing relations can be stored in a table. Index of the table is defined by the inputs to the routing relation, and the table entries correspond to the outputs.

For a particular network depending on the topology, the minimum paths from node a to node b will be defined by R-ab. E.g. for a torus network, the size of this min path set can then be found and a subset can be stored in a routing table. Note that by storing a subset, path diversity will be reduced and thus certain traffic patterns can perform poorer. Hence it’s important to minimize channel overlap among the subset to maximize diversity.

Table encoding

Tables can be design with all-at-once or incremental routing. Latter demands less localized storage.

Source table: used with all-at-once routing. Entire path is encoded in the route table. Storage is on the order of N number of nodes in the network. Additionally an alternative to storing a complete path per entry is to store a pointer into a long string of channels [1]. This encoding takes advantage of the fact that many paths may be subsets of longer paths.

[1] String is defined by cardinal direction of relative hops, e.g. hopping east twice and north once may be represented as ‘EEN’.

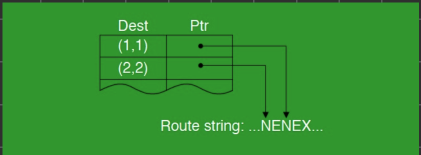


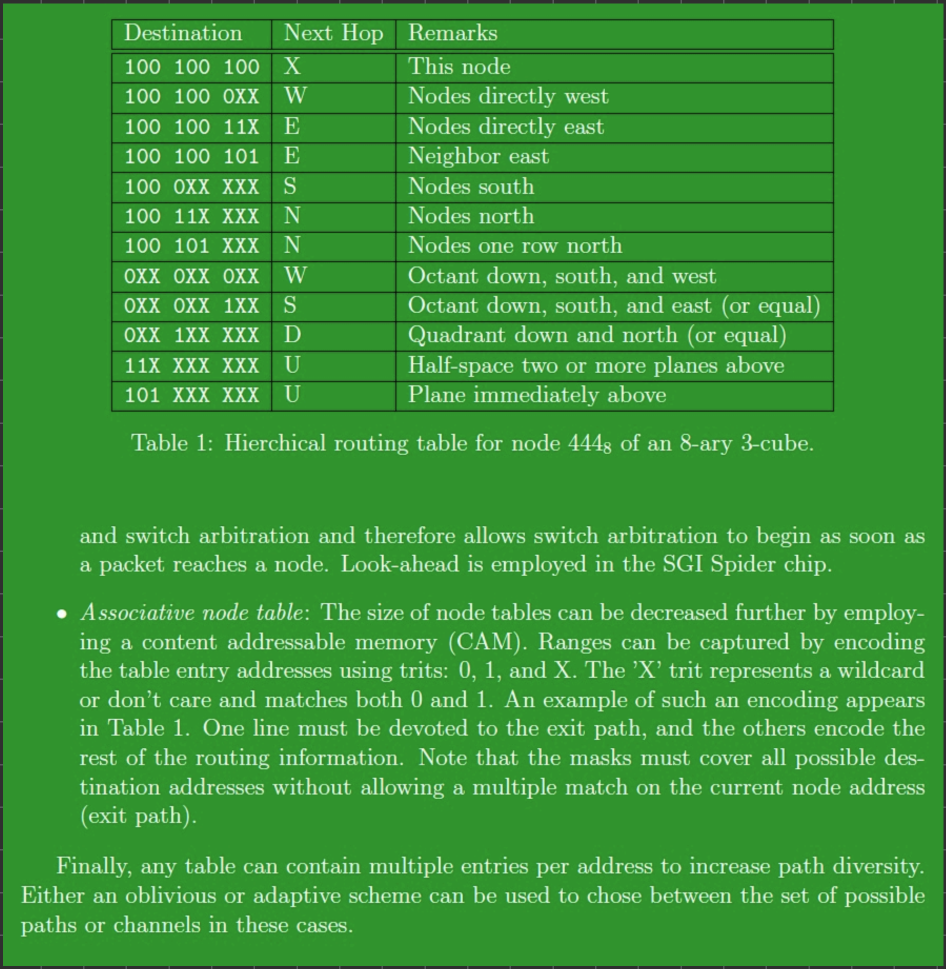
Figure 2: Encoding paths as pointers into a long route string

Presumably in this example, only a few route strings may have to be stored, and pointers can be used to index a subset of the respective min path

Node Table: similar principle to source table, however only paths to next hop are stored in routing table. Severely diminishes the flexibility of source table routing. E.g. once two paths destined to the same node share a link, paths cannot diverge. Source routing rectifies this.

> Based on this method, it’s evident that a node must lookup the next hop for each address. However a **look-ahead approach** can be used such that the packet carrier the next hop information, and the node performs a look-ahead for the next-next hop thereby pipelining switch arbitration with next-next-hop lookup thus reducing overall latency.

Associative node table: this technique further reduces local memory cost by using content addressable memory. Ranges are captured by using tristate digits: 0, 1, and X. See below for context.



Destination	Next Hop	Remarks
`100 100 100`	X	This node
`100 100 0XX`	W	Nodes directly west
`100 100 11X`	E	Nodes directly east
`100 100 101`	E	Neighbor east
`100 0XX XXX`	S	Nodes south
`100 11X XXX`	N	Nodes north
`100 101 XXX`	N	Nodes one row north
`0XX 0XX 0XX`	W	Octant down, south, and west
`0XX 0XX 1XX`	S	Octant down, south, and east (or equal)
`0XX 1XX XXX`	D	Quadrant down and north (or equal)
`11X XXX XXX`	U	Half-space two or more planes above
`101 XXX XXX`	U	Plane immediately above

Table 1: Hierchical routing table for node 444_8 of an 8-ary 3-cube.

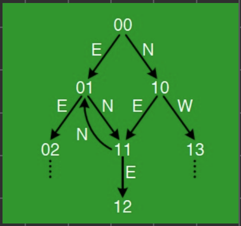
and switch arbitration and therefore allows switch arbitration to begin as soon as a packet reaches a node. Look-ahead is employed in the SGI Spider chip.

- *Associative node table*: The size of node tables can be decreased further by employing a content addressable memory (CAM). Ranges can be captured by encoding the table entry addresses using trits: 0, 1, and X. The ’X’ trit represents a wildcard or don’t care and matches both 0 and 1. An example of such an encoding appears in Table 1. One line must be devoted to the exit path, and the others encode the rest of the routing information. Note that the masks must cover all possible destination addresses without allowing a multiple match on the current node address (exit path).

Finally, any table can contain multiple entries per address to increase path diversity. Either an oblivious or adaptive scheme can be used to chose between the set of possible paths or channels in these cases.

Non-minimal routes

Can be used to aid in load balancing, however care must be taken to ensure progress in packets. If not properly addressed, a packet could be routed and then stuck in a loop as shown by the directed graph below.



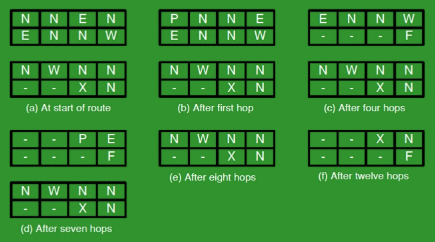
This is an example of a livelock, e.g. the packet continues hopping but never reaches the destination. (Note livelock is in contrast to deadlock where the packet gets stuck in a buffer due to issues in flow control.)

Additional state information can enforce a policy such that a maximum number of non minimal hops can be taken before the packet must be treated with strict minimal routing.

Route Encoding

When performing source routing, full path information must be transmitted in packet header. This information can be encoded using a N/S/E/W and X for the exit node.

In order to accommodate a scheme for arbitrary path length, we must use a phit continuation symbol and flit continuation symbol. See below



N	N	E	N
E	N	N	W

N	W	N	N
-	-	X	N

(a) At start of route

P	N	N	E
E	N	N	W

N	W	N	N
-	-	X	N

(b) After first hop

E	N	N	W
-	-	-	F

N	W	N	N
-	-	X	N

(c) After four hops

-	-	P	E
-	-	-	F

N	W	N	N
-	-	X	N

(d) After seven hops

N	W	N	N
-	-	X	N

(e) After eight hops

-	-	X	N
-	-	-	F

(f) After twelve hops

Figure 4: Arbitrary length encoding of source routes

Algorithmic routing => (Stub). Review “06 routing.pdf” for explanations.

Search based routing => (Stub). Review “06 routing.pdf” for explanations.