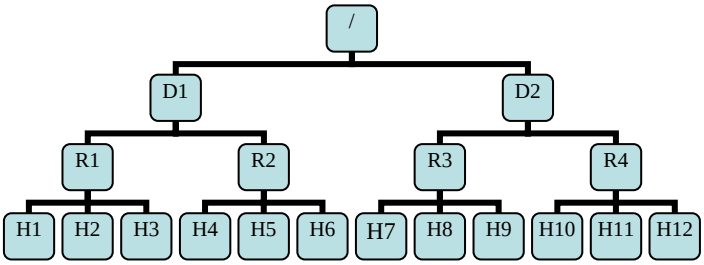
## *Assumption*




This issue assumes that HDFS runs on a cluster of computers with a tree hierarchical network topology. For example, a cluster may be consisted of many datacenters filled with racks of computers, as shown in the right figure, in which leaves represent computers and inner nodes represent switches/routers. Bandwidth in/out of a subtree may be less than the total bandwidth of machines within the subtree.

## *Network Topology*

Each node's position in the cluster is represented by a string with syntax similar to a file name. For example, R1's location is /D1/R1, while H8's location is /D2/R3/H8.

A data node gets its parent's location from the command line. The DataNode program supports an option "-p <id>" or "—parent < id>". If the option is not set, the data node belongs to a default parent. How to get a node's parent identifier is proprietary to each organization. So a mechanism needs to be provided when starting HDFS, for example, a script which prints the current machine's parent id on the screen. The output is then feed to data node when it starts.

The data node sends it's location to the name node as part of the registration information. Upon receiving a registration message, the name node first checks if the network topology already has an entry for the data node. If yes, it removes the old entry. It then adds the node. When the name node removes a data node, the data node entry in the cluster map is also removed.

The distance from a node to its parent is assumed to be 1. A distance between two computers can be calculated by summing up their distances to their closest common ancestor.

## *Replica placement*

The block replica placement policy is intended to get a tradeoff between minimizing the write cost and maximizing data reliability and availability, and aggregate read bandwidth.

When a new block is created, the first replica is placed on the local node, the second one is placed at a different rack, the third one is on a different node at the local rack, and the rest are placed on random nodes with restrictions that no more than one replica is placed at one node and no more than two replicas are placed in the same rack when the number of replicas is less than twice of its rack number.

When re-replicating a block, if the number of existing replicas is one, place the second one on a different rack. When the number of existing replicas is two, if the two replicas are on the same rack, place the third one on a different rack; otherwise, place the third one on a different node at the same rack of first replica. When the number of available replicas is more than two, place the rest of the replicas randomly.

After all target nodes are selected, nodes are organized as a pipleline in the order of their closeness to the first replica. Data are forwarded to nodes in this order.

For reading, the name node first checks if the client’s computer is located in the cluster. If yes, block locations are returned to the client in the order of its closeness to the reader. The block is read from data nodes in this preference order.