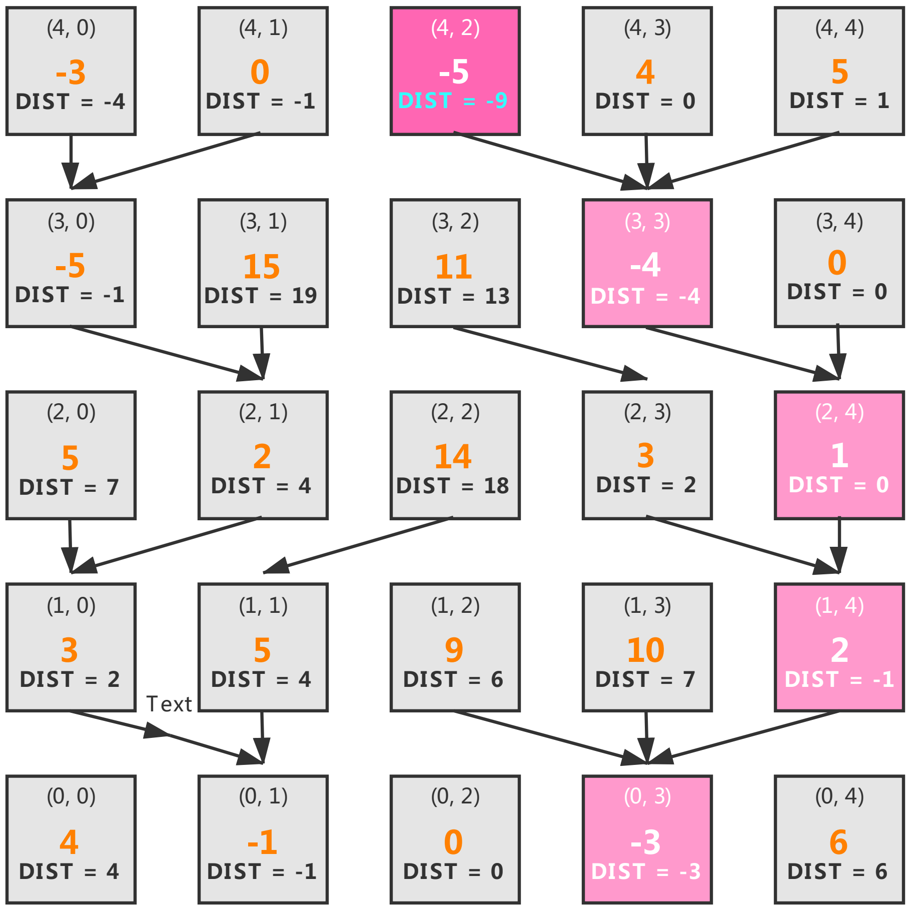


Full Intelligence

NODE

last	:	Node
cost	:	Int
dist	:	Int

Conclusion:

Since Each node of each row always holds the shortest path from the bottom, each node of the top layer will ultimately holds the shortest route from bottom to themselves. Thus, by finding the node of top layer with the shortest DIST, we can get the global shortest path.

Each node of each layer starting from the 2nd row at the bottom will hold the reference of a reachable node below them with the least distance.

These node will set their DISTANCE to the sum of COST of themselves and the DISTANCE of the last node recursively. And nodes of 1st layer will set their DISTANCE to their COST, and reference to null.

Node[i][j].last = MinDist(Node[i-1][j], Node[i-1][j-1], Node[i-1][j+1]);
i > 0;

Node[i][j].dist = Node[i][j].cost + Node[i][j].last.cost;

Node[0][j].dist = Node[0][j].cost;

(Node with j+1 of j-1 may not exist, if so, just ignore them.)

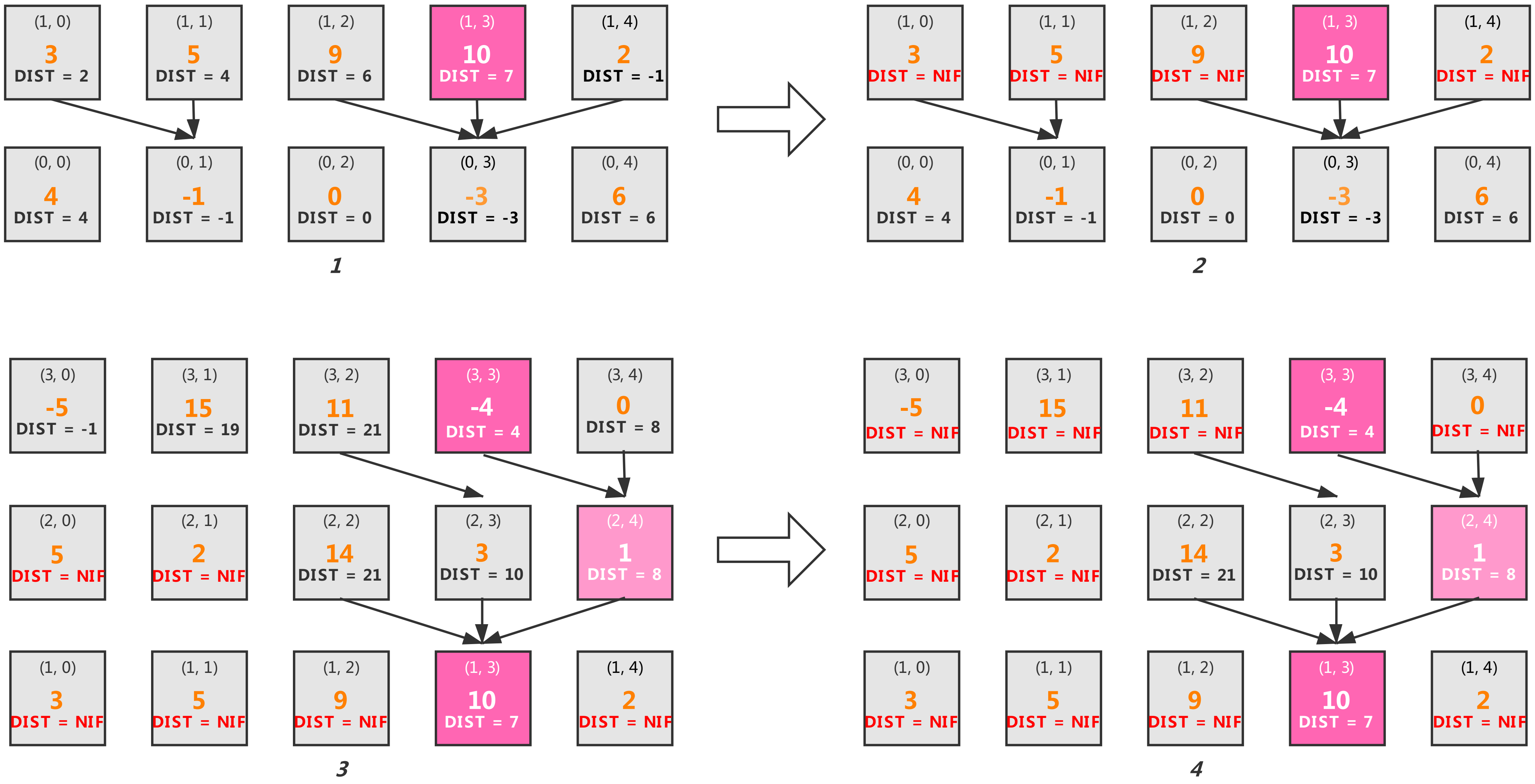
When the INTELLIGENCE is below 100%, the algorithm will find the node with shortest distance within limited rows.

Like the example above, since intelligence is set to 40%, the algorithm can only make decision within 2 rows.

In order to make sure nodes without the lowest distance being selected on the next time algorithm runs, we simply set them to infinity, so no one would pick these route up in the end, since their distance are extremely long.

Conclusion:

So in most of time, lower intelligence will lead to a worse result.



40% Intelligence