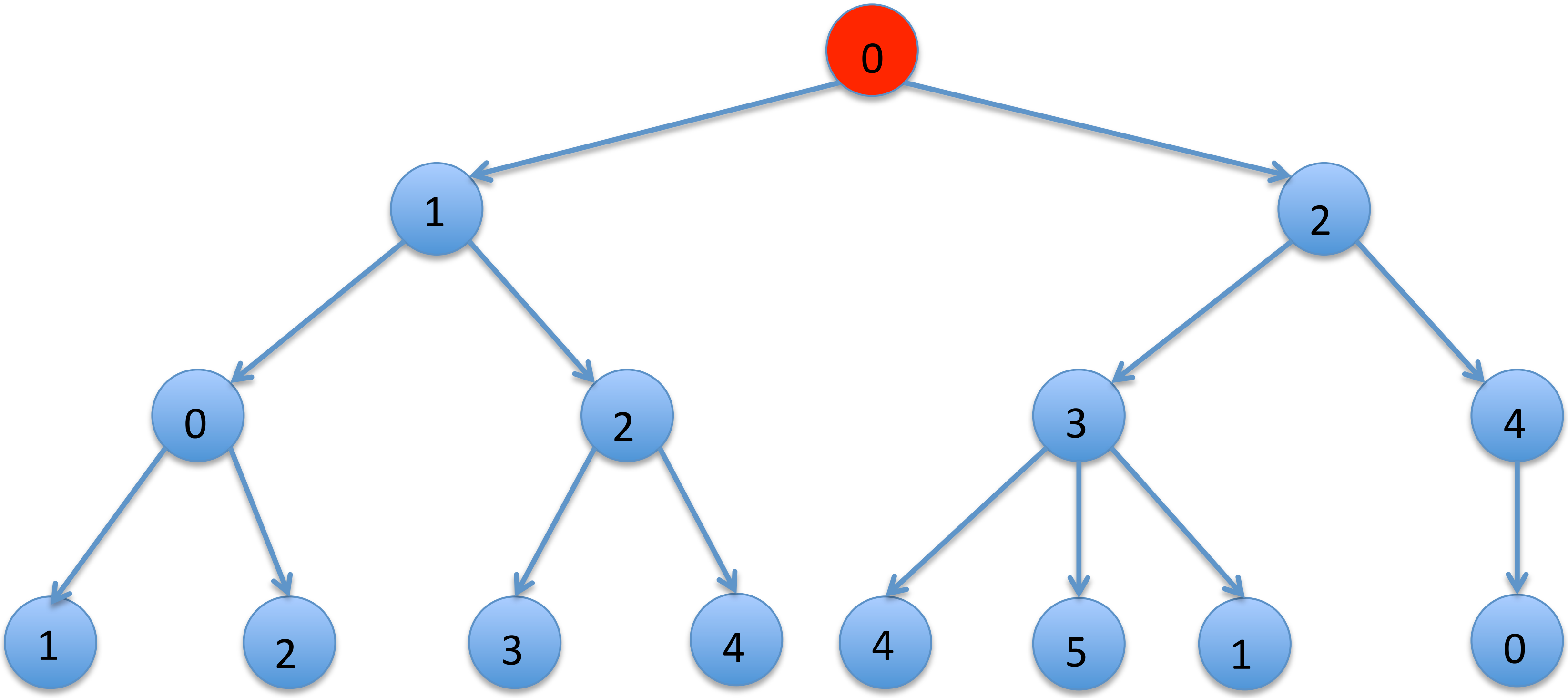
To find shortest path

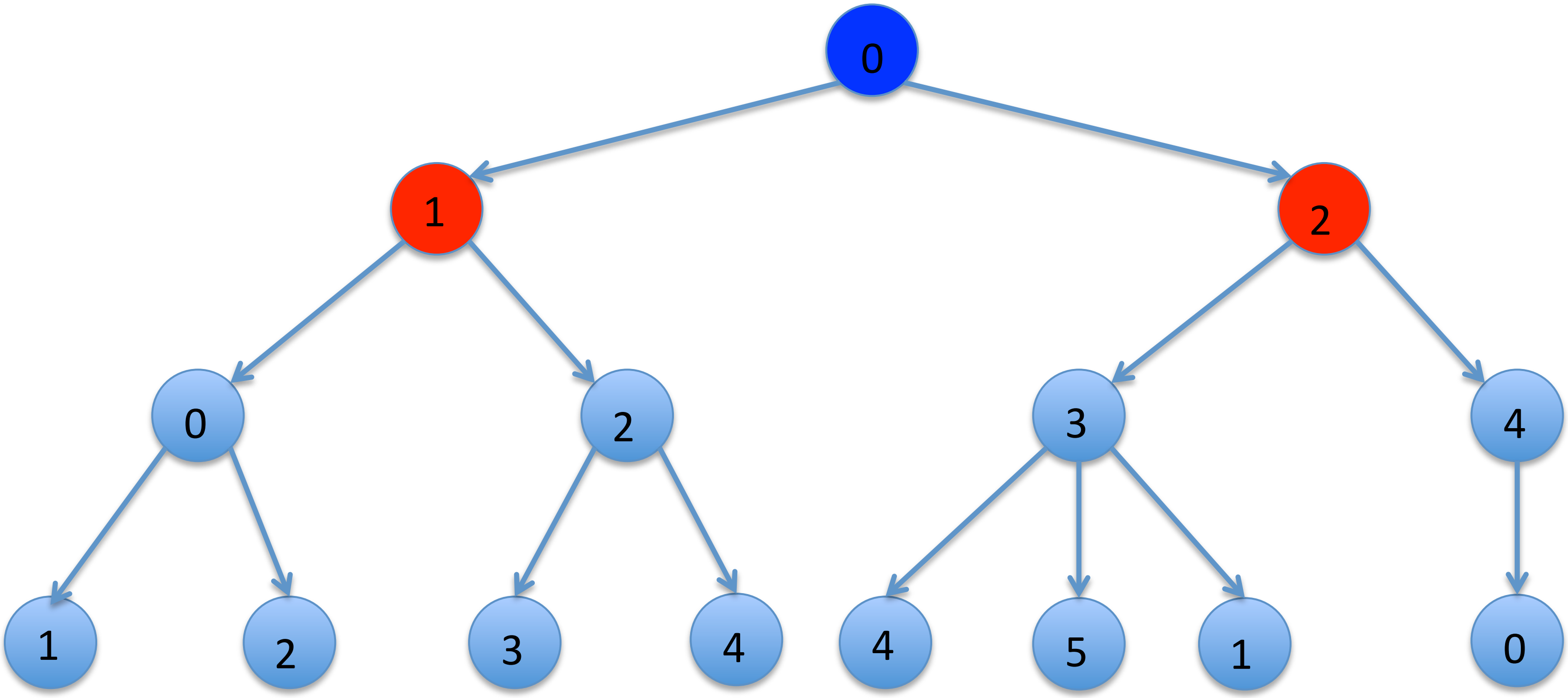
- We need to keep track of the best path found so far
- When we find a new path, keep going only if path still shorter than best seen so far

Better depth first search



0

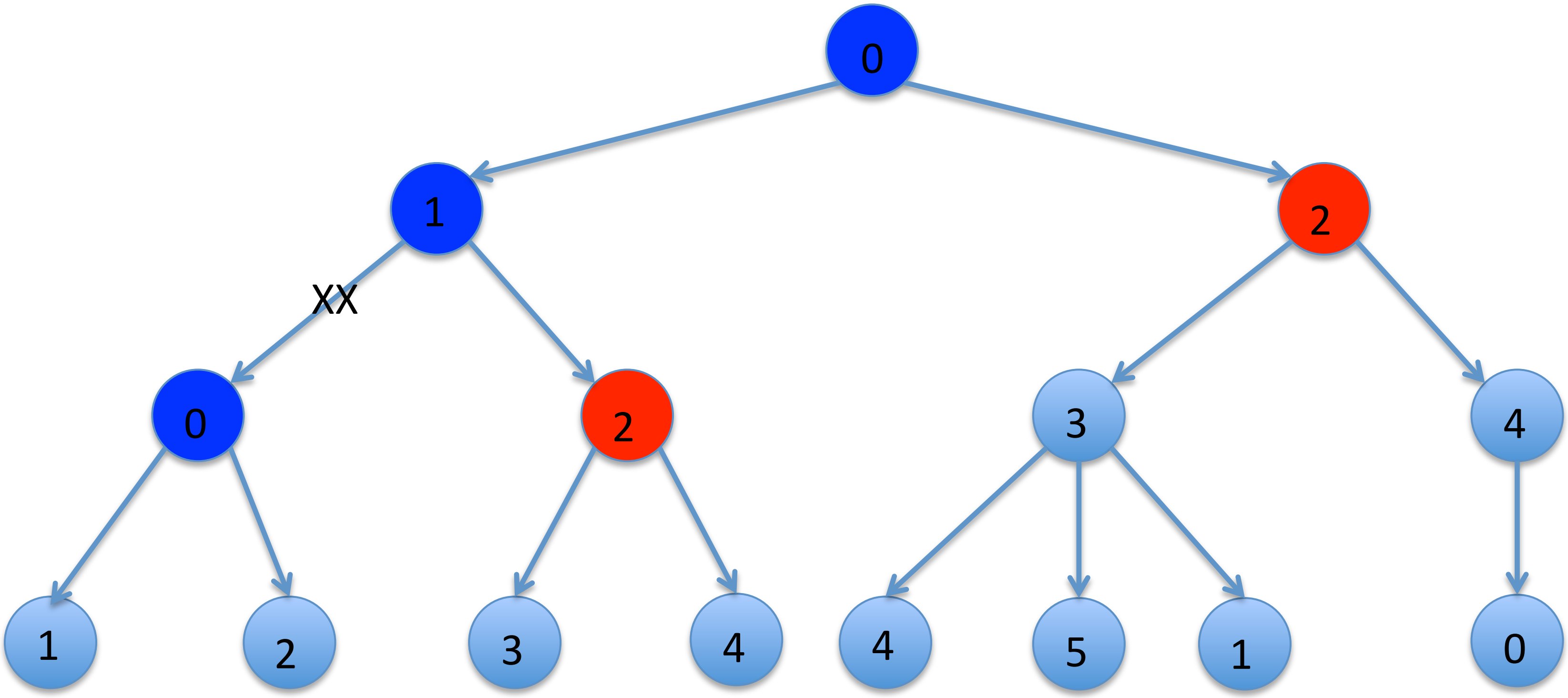
Better depth first search



0

01 02

Simple depth first search

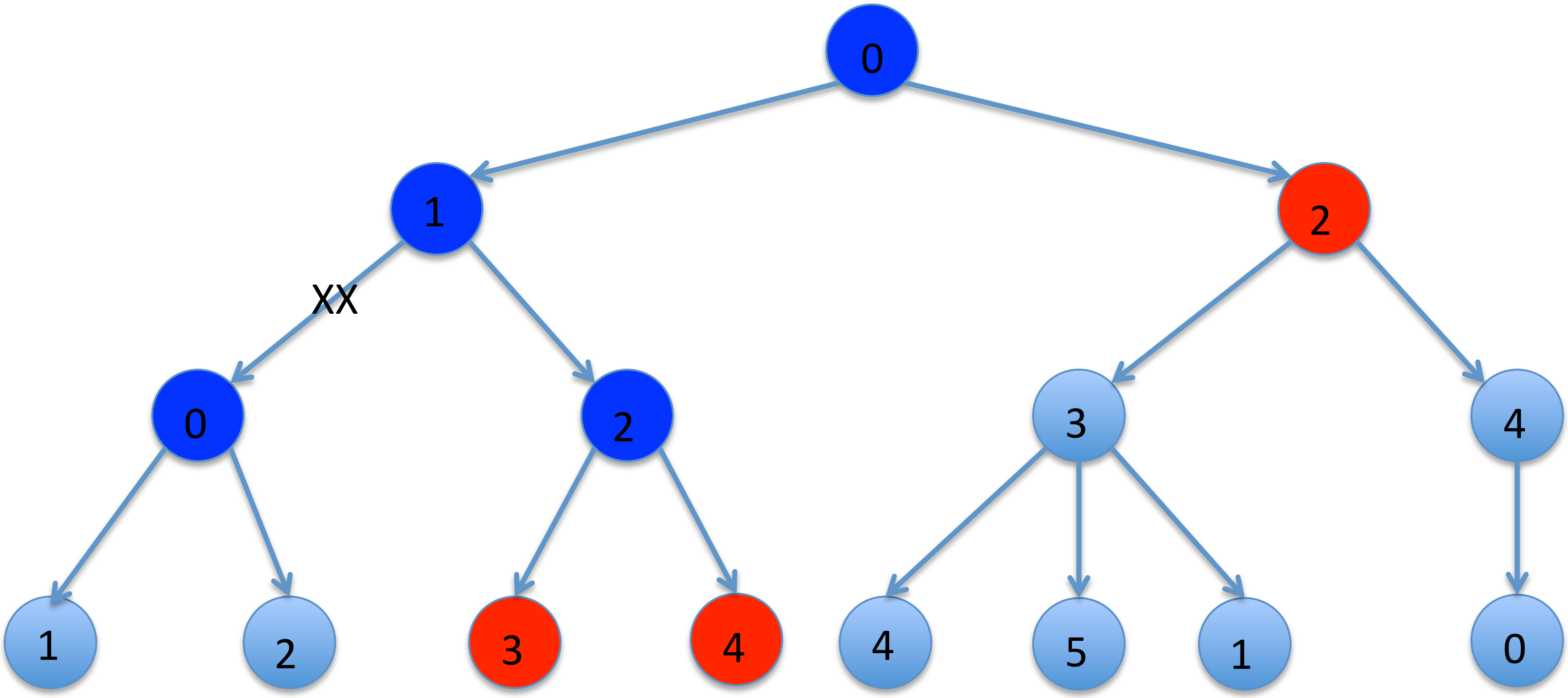


0

01 02

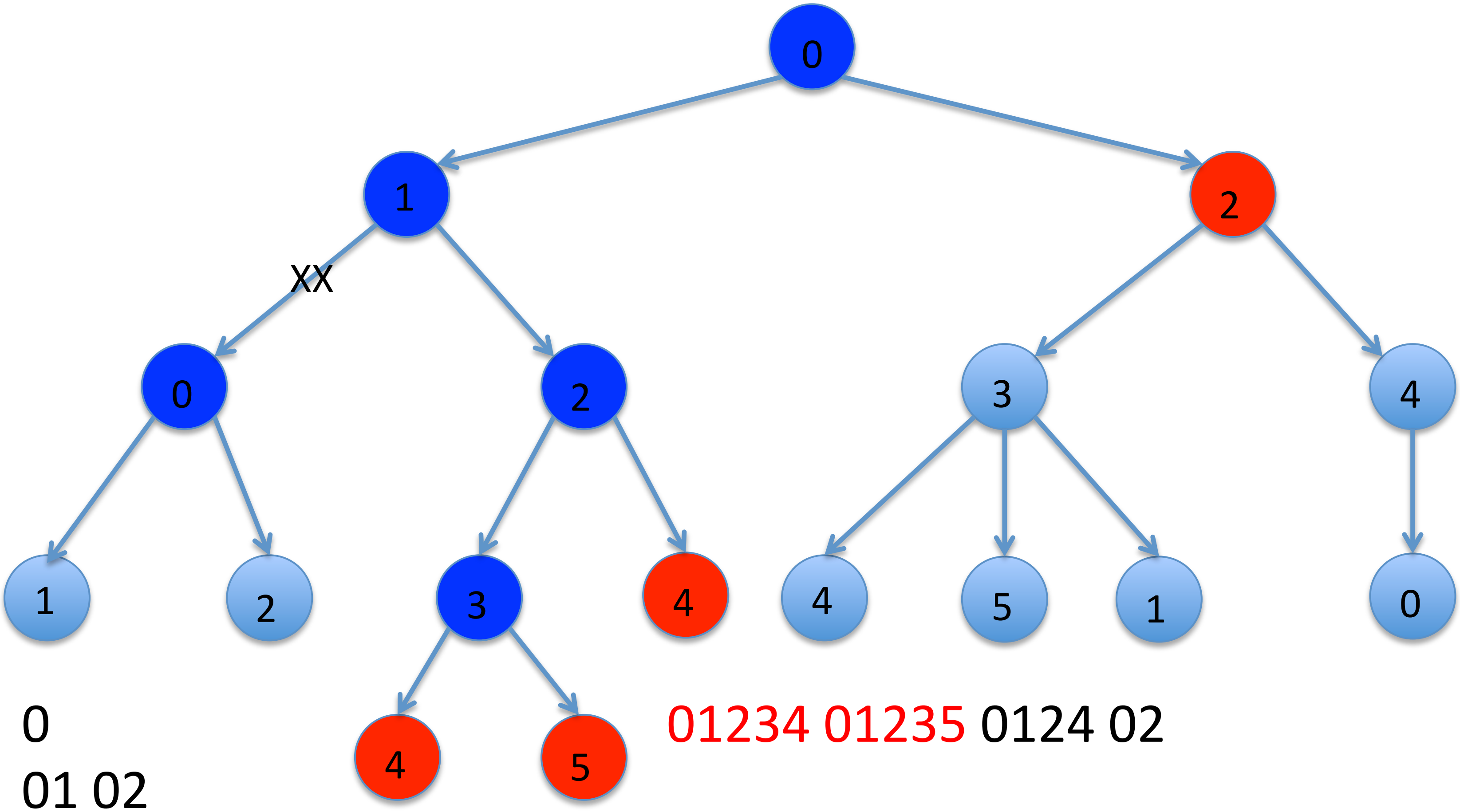
012 02

Simple depth first search



0
01 02
012 02
0123 0124 02

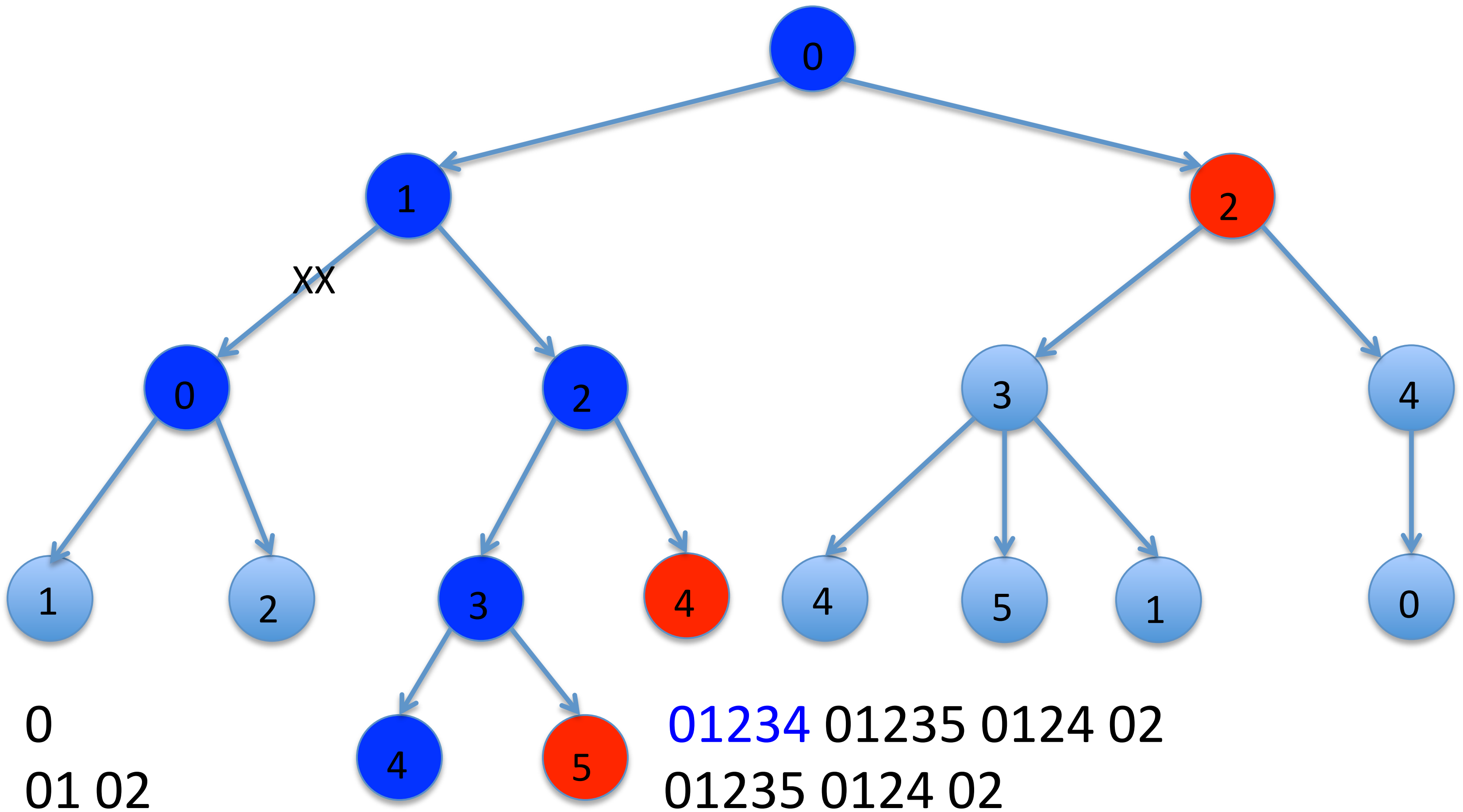
Simple depth first search



01234 01235 0124 02

0
01 02
012 02
0123 0124 02

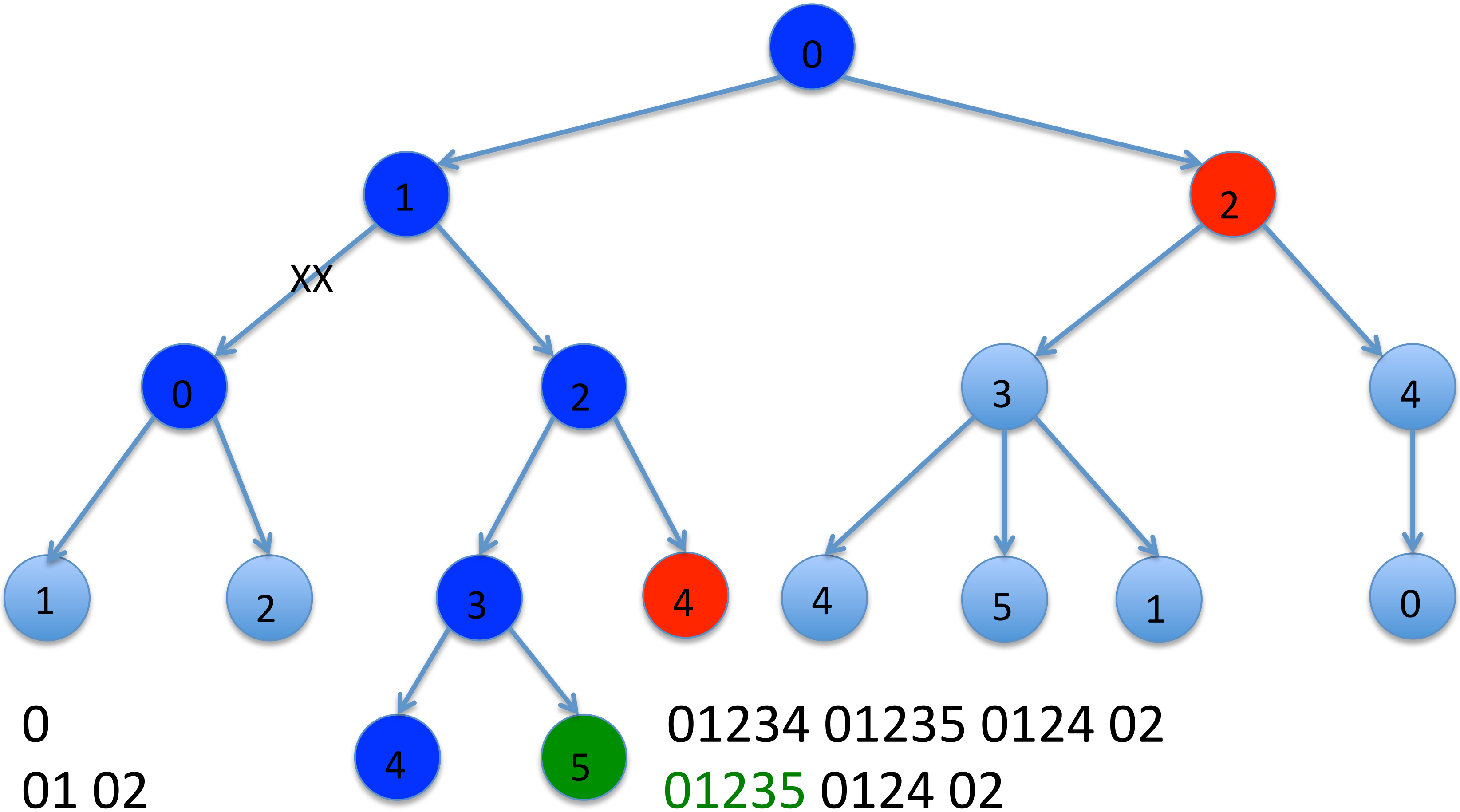
Simple depth first search



0
01 02
012 02
0123 0124 02

01234 01235 0124 02
01235 0124 02

Simple depth first search



0

01 02

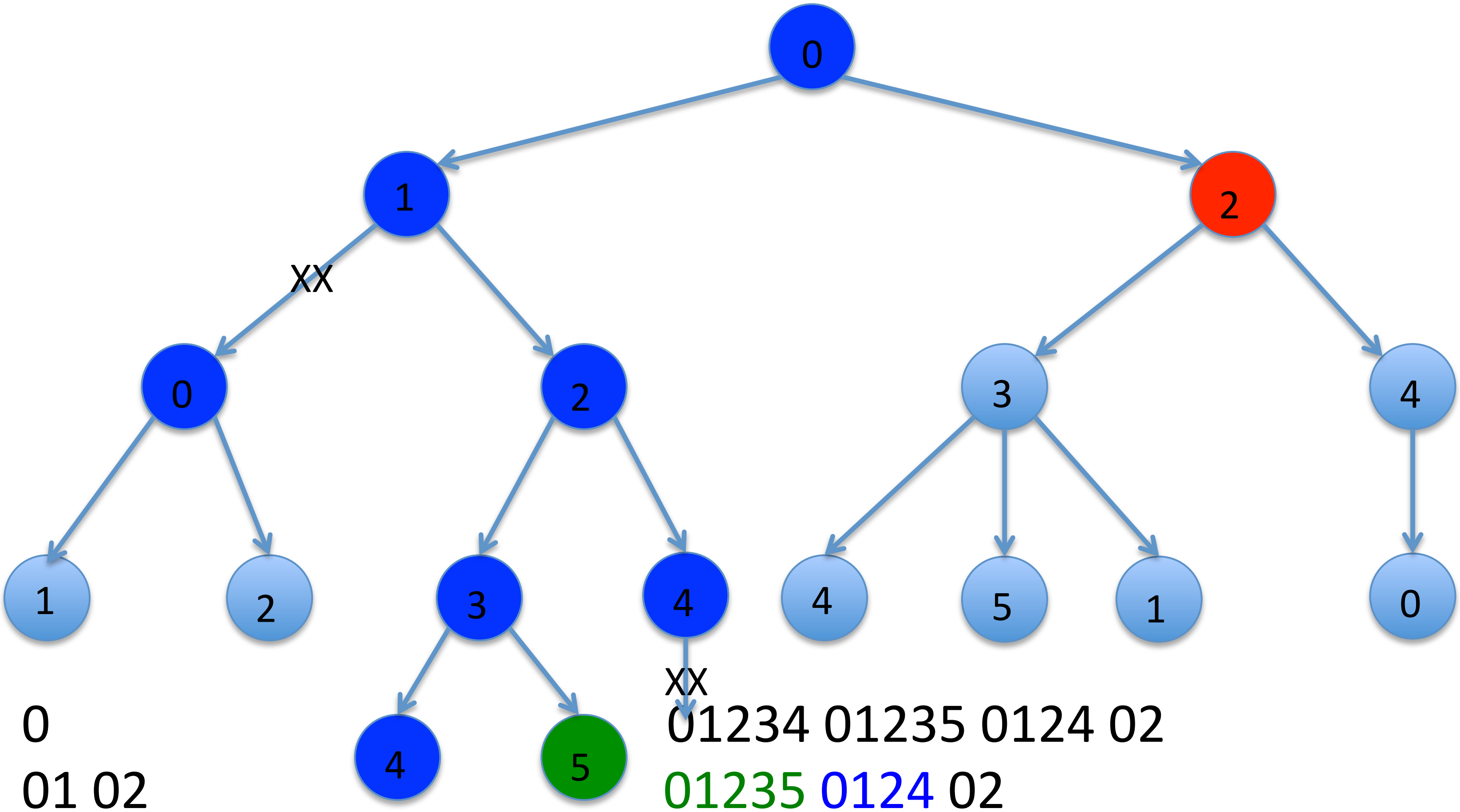
012 02

0123 0124 02

01234 01235 0124 02

01235 0124 02

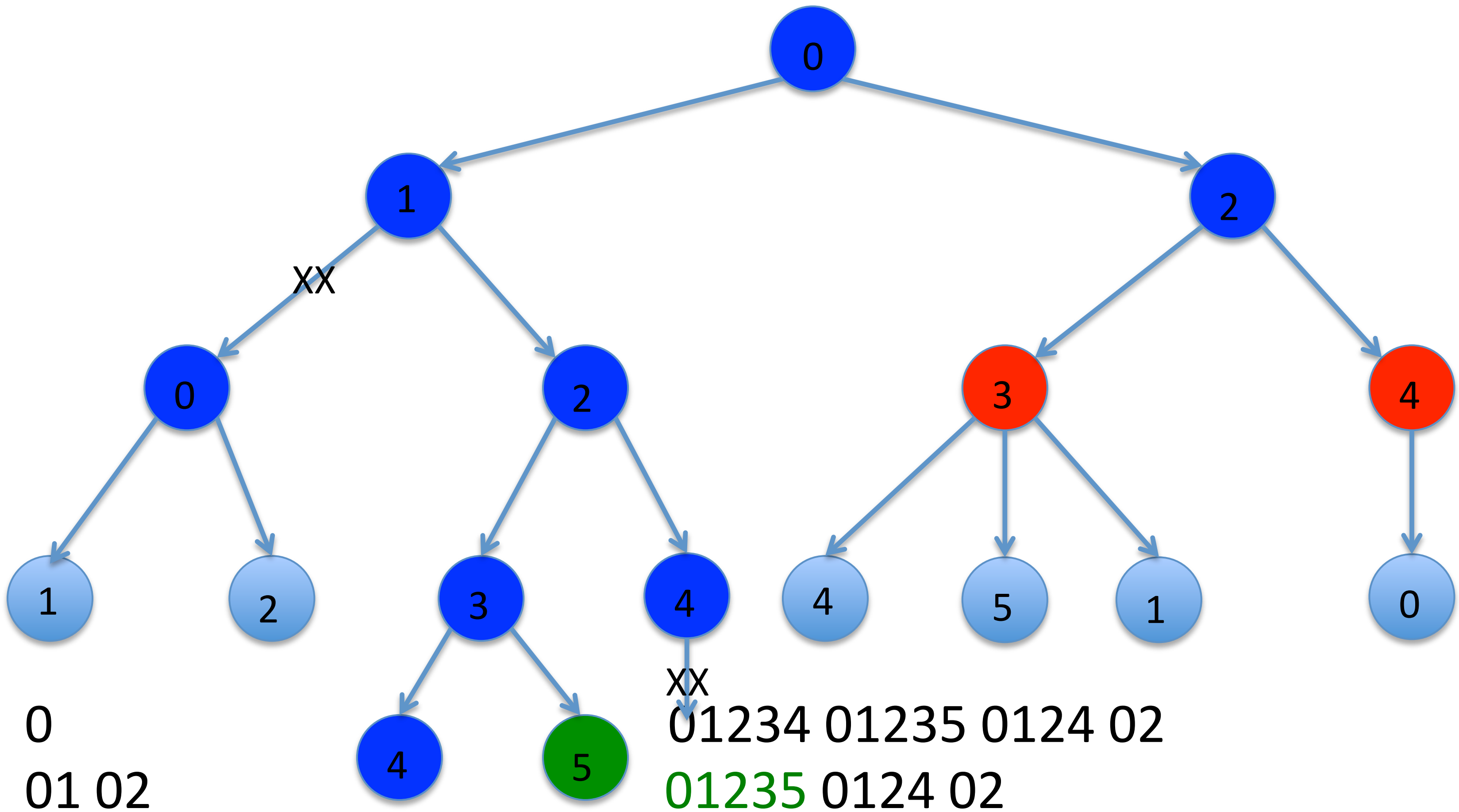
Simple depth first search



0
01 02
012 02
0123 0124 02

01234 01235 0124 02
01235 0124 02
02

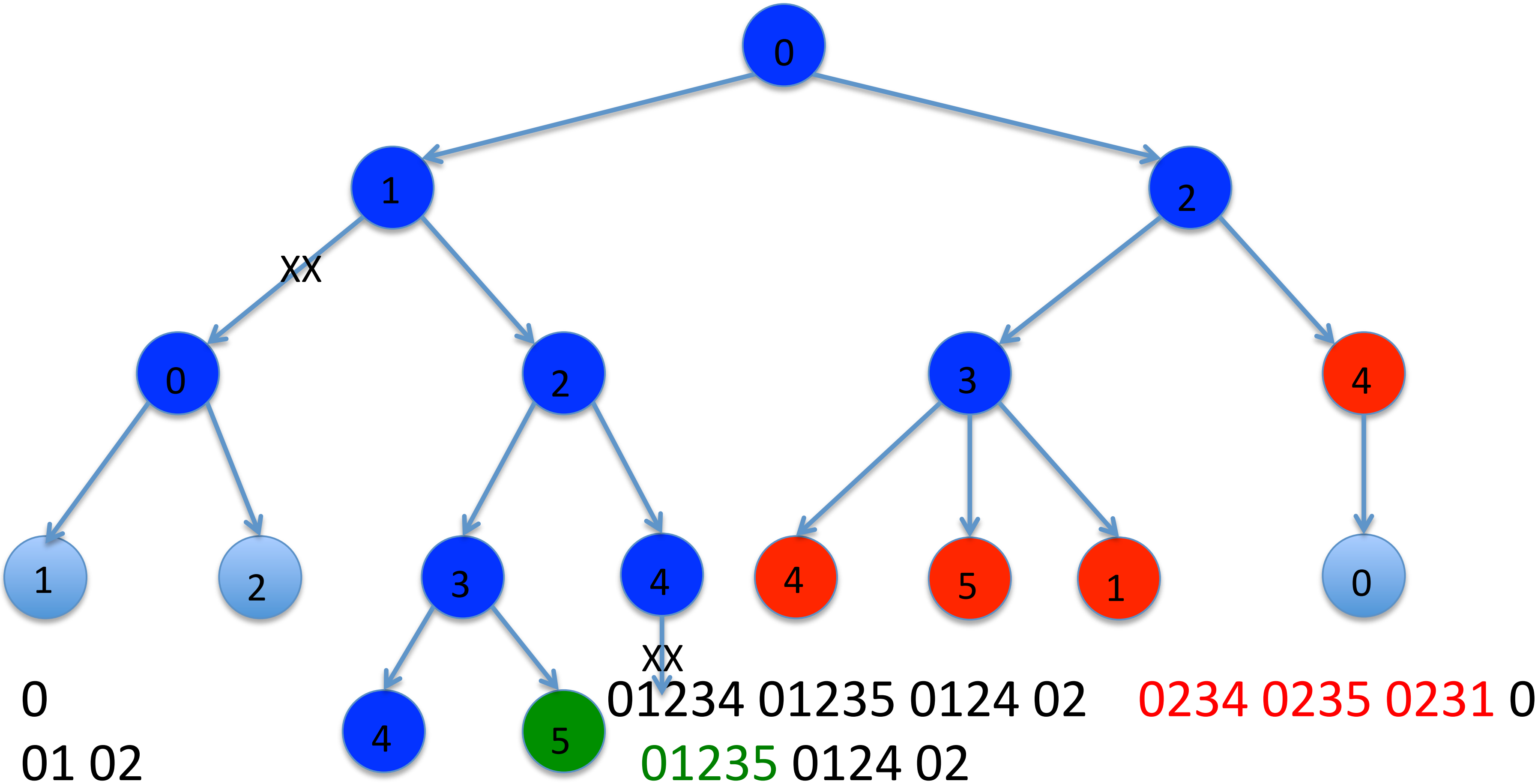
Simple depth first search



0
01 02
012 02
0123 0124 02

01234 01235 0124 02
01235 0124 02
02
023 024

Simple depth first search



0

01 02

012 02

0123 0124 02

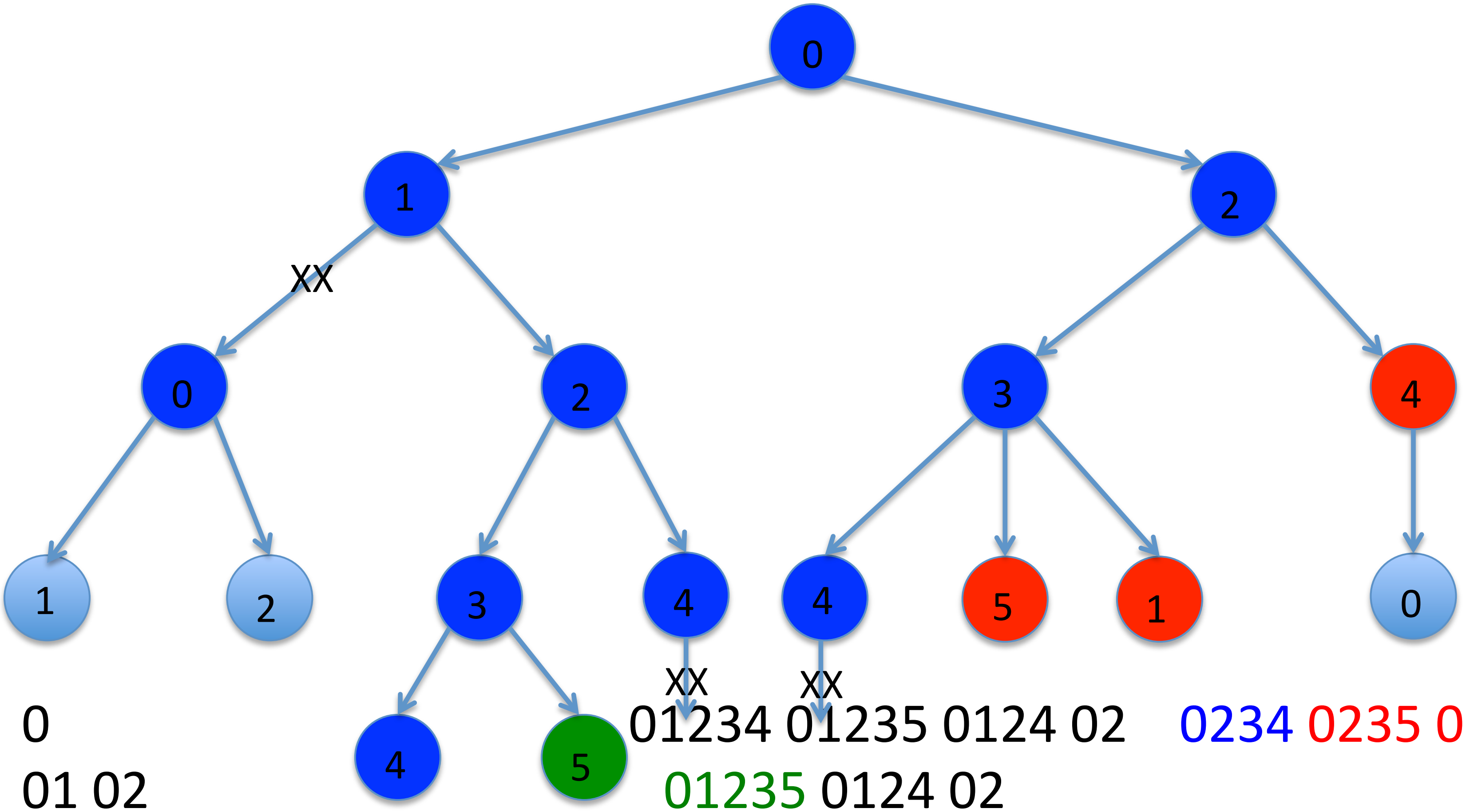
01234 01235 0124 02   0234 0235 0231 024

01235 0124 02

02

023 024

Simple depth first search



0

01 02

012 02

0123 0124 02

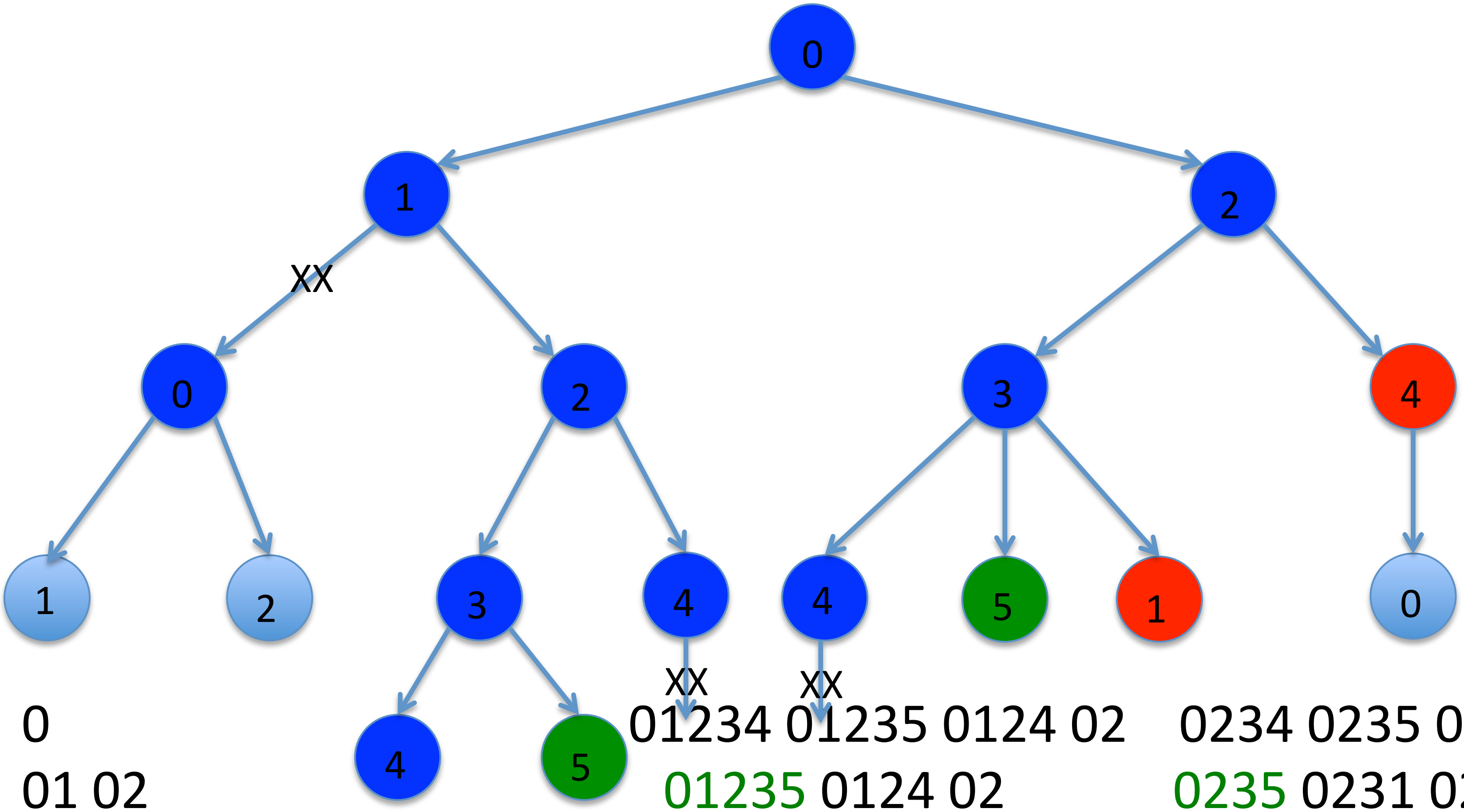
01234 01235 0124 02   0234 0235 0231 024

01235 0124 02

02

023 024

Simple depth first search



0

01 02

012 02

0123 0124 02

01234 01235 0124 02

01235 0124 02

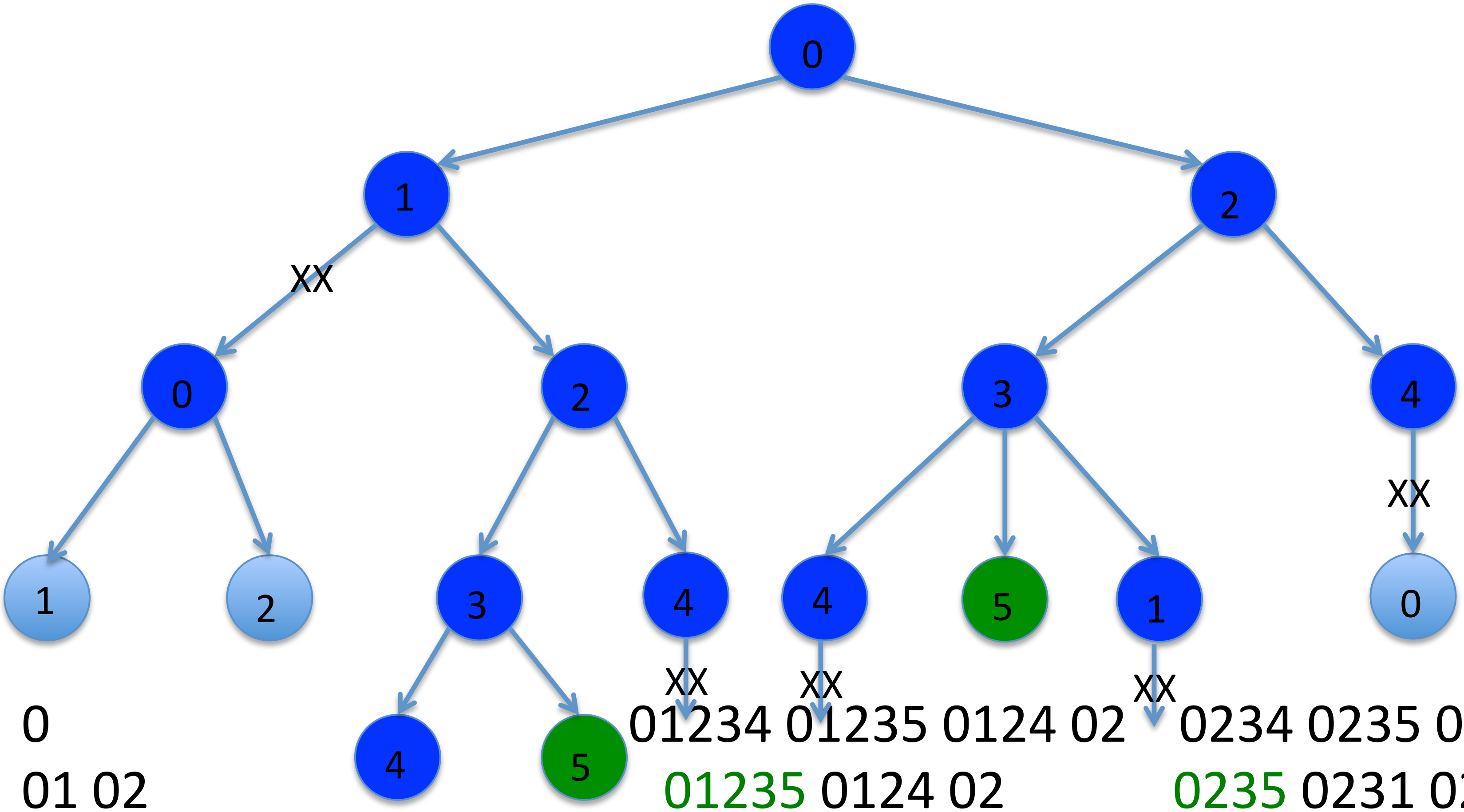
02

023 024

0234 0235 0231 024

0235 0231 024

Simple depth first search

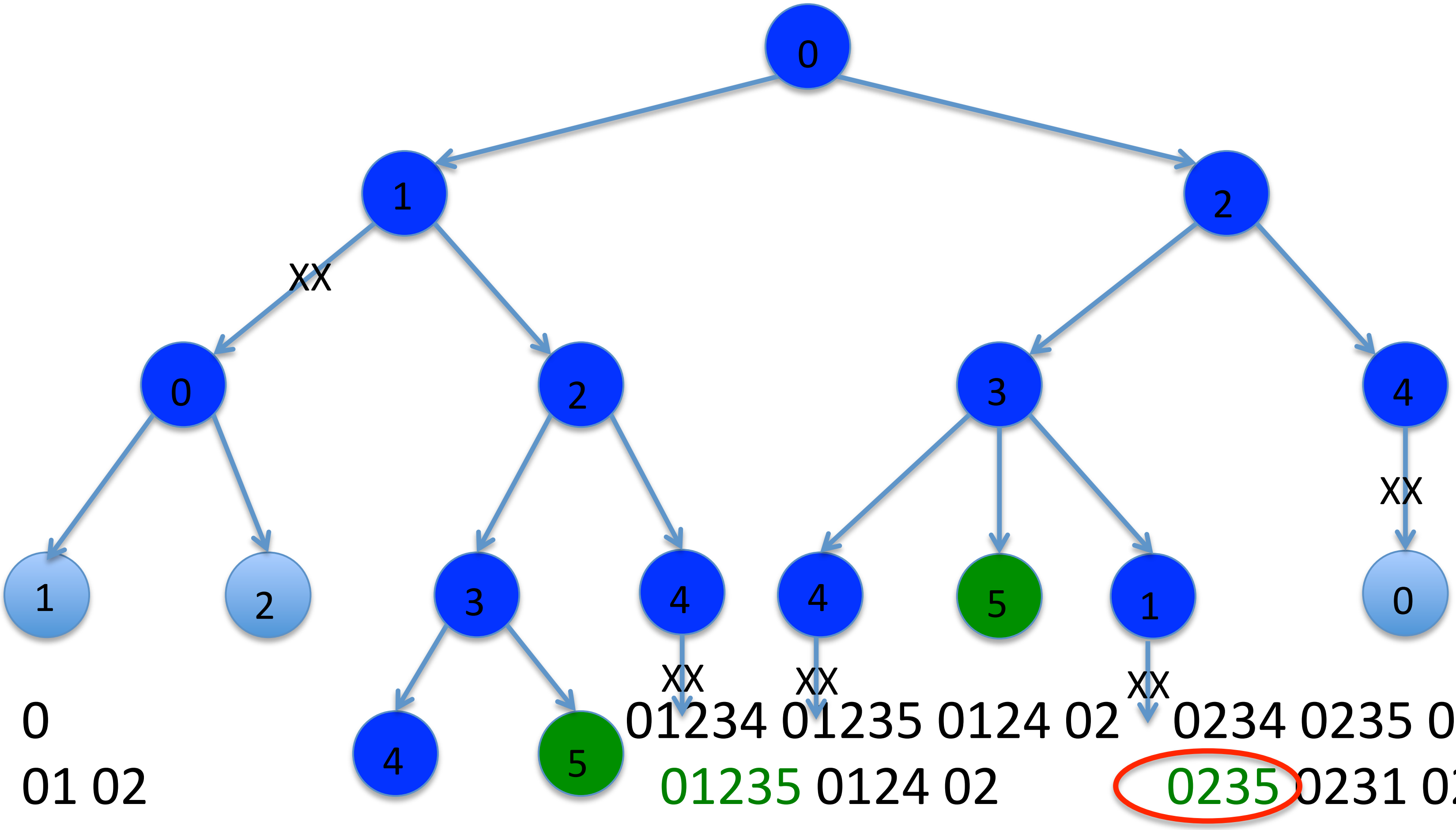


0
01 02
012 02
0123 0124 02

01234 01235 0124 02
01235 0124 02
02
023 024

0234 0235 0231 024
0235 0231 024
024

Simple depth first search



0
01 02
012 02
0123 0124 02

01234 01235 0124 02
01235 0124 02
02
023 024

0234 0235 0231 024
0235 0231 024
024

A shortest path DFS algorithm

```
def DFS(graph, start, end, path = [],shortest = None):
    # Assumes graph is a Digraph
    # Assumes start and end are nodes in graph
    path = path + [start]
    print 'Current dfs path:', printPath(path)
    if start == end:
        return path
    for node in graph.childrenOf(start):
        if node not in path: # Avoid cycles
            if shortest == None or len(path) < len(shortest):
                newPath = DFS(graph, node, end, path, shortest)
                if newPath != None:
                    shortest = newPath
    return shortest
```