Clone Graph

Got it! Let's carefully explain this DFS cloning of a graph step by step with this example.

✅ The Graph

We have an undirected graph:

```lua
1 -- 2
|    |
4 -- 3
```

- Node `1` connects to 2 and 4
- Node `2` connects to 1 and 3
- Node `3` connects to 2 and 4
- Node `4` connects to 1 and 3

We want to clone this graph, meaning:

- Create a new copy of every node
- Make sure new nodes have the same connections (neighbors)
- Avoid infinite loops (since graph has cycles!)

✅ How DFS cloning works

We use a map (dictionary) called `cloned` to remember:

- If a node is already cloned
- What its cloned copy is

When we visit a node:

1. If it's not cloned yet, we create a new node.
2. Then we DFS its neighbors:
   - If neighbor is cloned → just return it
   - If not cloned → recursively clone it
3. Attach cloned neighbors to the cloned node.

✅ Step-by-step cloning

We start cloning from node1.

Step 1 → Clone node1

- `cloned` is empty → so create a new node copy1(val=1)
- `cloned[1] = copy1`
- Now go through node1's neighbors → `[node2, node4]`

Step 2 → Clone node2 (neighbor of node1)

- node2 is not cloned yet → create copy2(val=2)
- `cloned[2] = copy2`
- Now go through node2's neighbors → `[node1, node3]`

Neighbor of node2 → node1

- node1 is already cloned → return `copy1`

Neighbor of node2 → node3

- node3 is not cloned yet → so clone it

Step 3 → Clone node3 (neighbor of node2)

- node3 is not cloned yet → create copy3(val=3)
- `cloned[3] = copy3`
- Now go through node3's neighbors → `[node2, node4]`

Neighbor of node3 → node2

- node2 is already cloned → return `copy2`

Neighbor of node3 → node4

- node4 is not cloned yet → so clone it

Step 4 → Clone node4 (neighbor of node3)

- node4 is not cloned yet → create copy4(val=4)
- `cloned[4] = copy4`
- Now go through node4's neighbors → `[node1, node3]`

Neighbor of node4 → node1

- already cloned → return `copy1`

Neighbor of node4 → node3

- already cloned → return `copy3`

Now node4's neighbors are done.

✅ Backtracking…

- Now node3 gets its neighbors `[copy2, copy4]`
- Now node2 gets its neighbors `[copy1, copy3]`
- Now node1 gets its neighbors `[copy2, copy4]`

✅ Final cloned graph

The cloned graph is exactly like the original:

lua Copy Edit

```
copy1 -- copy2
  |       |
copy4 -- copy3
```

- No cycles issue because we used `cloned` dictionary to avoid cloning again.

✅ Why do we need `cloned`?

If we didn't keep track, we'd keep cloning forever because the graph has cycles:

- node1 → node2 → node3 → node4 → back to node1…

So `cloned` acts like a visited set + map to cloned nodes.

Would you like me to:

✅ Draw a visual diagram of how DFS expands step by step?
✅ Or show you the actual DFS clone code in Python with comments?