1136. 并行课程

ITCharge 大约 2 分钟

- 标签：图、拓扑排序
- 难度：中等

题目链接

- 1136. 并行课程 - 力扣

题目大意

有 N 门课程，分别以 1 到 N 进行编号。现在给定一份课程关系表 `relations[i] = [X, Y]`，用以表示课程 `X` 和课程 `Y` 之间的先修关系：课程 `X` 必须在课程 `Y` 之前修完。假设在一个学期里，你可以学习任何数量的课程，但前提是你已经学习了将要学习的这些课程的所有先修课程。

要求：返回学完全部课程所需的最少　　数。如果没有办法做到学完全部这些课程的话，就返回 `-1` 。

解题思路

拓扑排序。具体解法如下：

1. 使用列表 `edges` 存放课程关系图，并统计每门课程节点的入度，存入入度列表 `indegrees` 。使用 `ans` 表示学期数。
2. 借助队列 `queue` ，将所有入度为 `0` 的节点入队。
3. 将队列中所有节点依次取出，学期数 +1。对于取出的每个节点：
    1. 对应课程数 -1。
    2. 将该顶点以及该顶点为出发点的所有边的另一个节点入度 -1。如果入度 -1 后的节点入度不为 0，则将其加入队列 `queue` 。
4. 重复 3~4 的步骤，直到队列中没有节点。
5. 最后判断剩余课程数是否为 0，如果为 0，则返回 `ans` ，否则，返回 `-1` 。

代码

```py
import collections

class Solution:
    def minimumSemesters(self, n: int, relations: List[List[int]]) -> int:
        indegrees = [0 for _ in range(n + 1)]
        edges = collections.defaultdict(list)
        for x, y in relations:
            edges[x].append(y)
            indegrees[y] += 1
        queue = collections.deque([])
        for i in range(1, n + 1):
            if not indegrees[i]:
                queue.append(i)
        ans = 0

        while queue:
            size = len(queue)
            for i in range(size).
                x = queue.popleft()
                n -= 1
                for y in edges[x]:
                    indegrees[y] -= 1
                    if not indegrees[y]:
                        queue.append(y)
            ans += 1

        return ans if n == 0 else -1
```