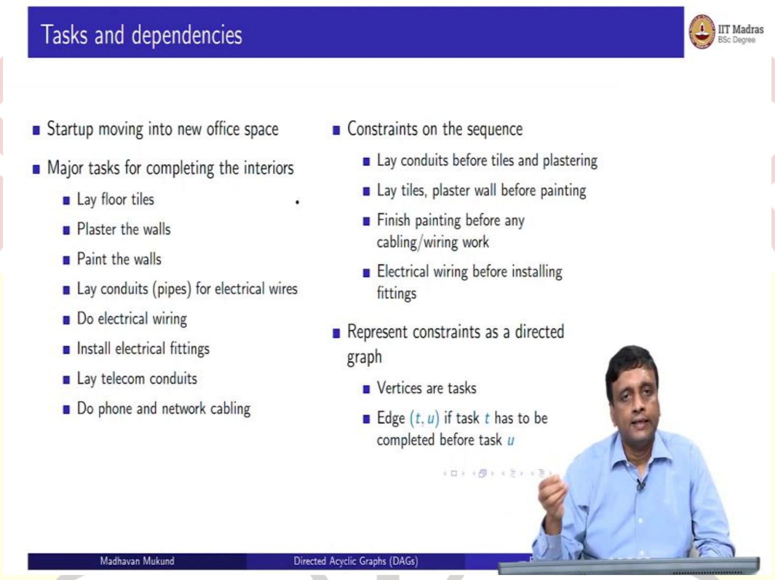
IIT Madras

ONLINE DEGREE

Programming Data Structures and Algorithms using Python
Professor. Madhavan Mukund
Directed Acyclic Graphs (DAGs)

So, we saw that we can use BFS and DFS to detect cycles in a graph. In an undirected graph, we just look for non-tree edges and in a directed graph, we use the DFS numbering in order to look for back edges and find cycles. Now, there are actually a large class of applications where directed at acyclic graphs are important.

(Refer Slide Time: 00:30)



Tasks and dependencies

- Startup moving into new office space
- Major tasks for completing the interiors
  - Lay floor tiles
  - Plaster the walls
  - Paint the walls
  - Lay conduits (pipes) for electrical wires
  - Do electrical wiring
  - Install electrical fittings
  - Lay telecom conduits
  - Do phone and network cabling
- Constraints on the sequence
  - Lay conduits before tiles and plastering
  - Lay tiles, plaster wall before painting
  - Finish painting before any cabling/wiring work
  - Electrical wiring before installing fittings
- Represent constraints as a directed graph
  - Vertices are tasks
  - Edge (t, u) if task t has to be completed before task u

Madhavan Mukund Directed Acyclic Graphs (DAGs)

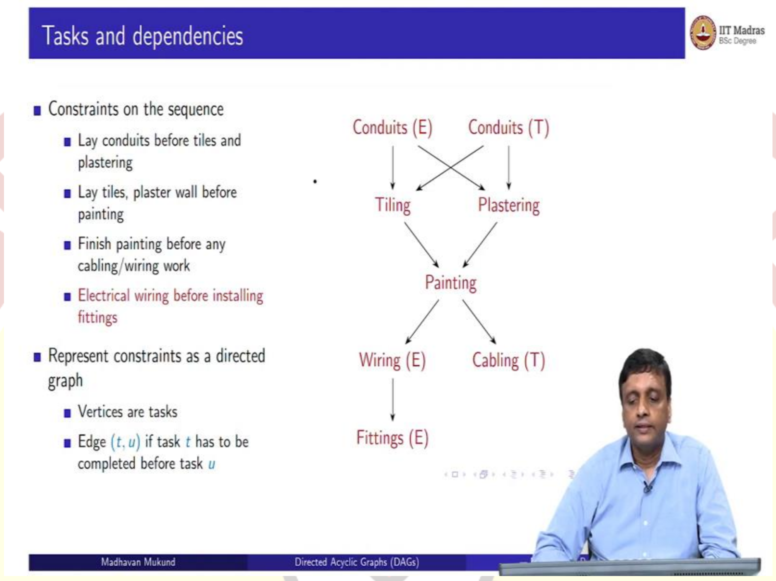
Typically, a directed acyclic graph is what you need to represent a set of tasks and their dependencies. So, suppose you are moving into a new office, let us say a start-up is moving to a new office. Now, there are a number of things that need to be done to make the office space ready for occupation.

Do you have to lay the tiles, plaster the walls, paint the walls, you also have to lay conduits because there are going to be lots of cables and wires, you have to do the electrical wiring you have to do the electrical fittings, the plug points, but you also have to do the telecom wiring and the telecom cabling, so all this has to be done and there will be some constraints usually over this.

So, there will be some constraints. Let us say for instance, that you obviously cannot break open the wall after it is plastered and painted, not really conduit, so conduits have to be put before tiles are put on the floor before the plastering is done, you would probably want to lay the tiles and plaster the wall before you paint, because especially Of course, you have to plaster the wall before you paint. But then if you put tiles after you painted the walls, some of the cement used to lay the tiles might splash on the walls.

And finally, you might want to finish painting before you do some cabling and wiring and you might want to do the electric wiring before you do this fitting because you cannot put the light bulbs refill wires are not in place. So, these are typically represented as a directed graph in which the vertices are the tasks to be done and there is a dependency. So, these constraints say you must do this before you do that. So, that is a directed edge u before v.

(Refer Slide Time: 02:05)



Tasks and dependencies

- Constraints on the sequence
  - Lay conduits before tiles and plastering
  - Lay tiles, plaster wall before painting
  - Finish painting before any cabling/wiring work
  - Electrical wiring before installing fittings
- Represent constraints as a directed graph
  - Vertices are tasks
  - Edge (t, u) if task t has to be completed before task u

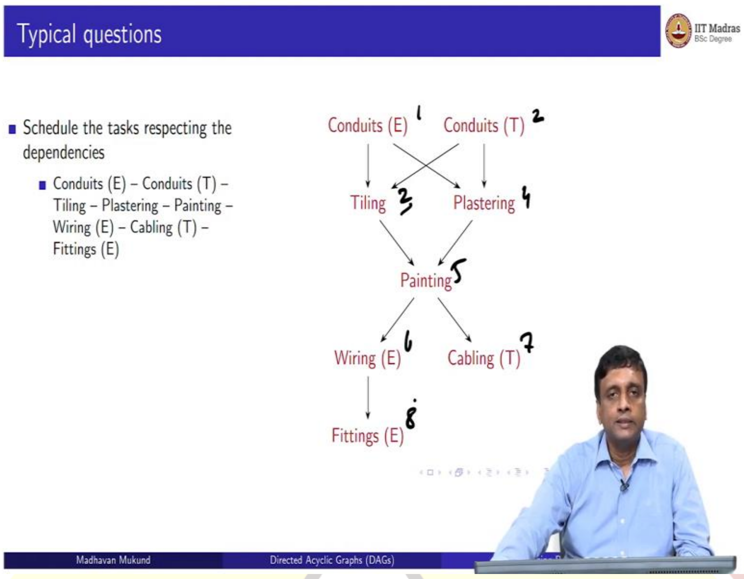
$$\begin{tikzcd}
\text{Conduits (E)} \arrow[d] \arrow[dr] & \text{Conduits (T)} \arrow[d] \arrow[dl] \\
\text{Tiling} \arrow[dr] & \text{Plastering} \arrow[d] \\
& \text{Painting} \arrow[dl] \arrow[d] \\
\text{Wiring (E)} \arrow[d] & \text{Cabling (T)} \\
\text{Fittings (E)} &
\end{tikzcd}$$

Madhavan Mukund Directed Acyclic Graphs (DAGs)

So, let us look at these constraints. So, we have these tasks. So, these are the vertices. So, here is the graph that we might want to construct using these vertices. So, we have these various tasks, you know, conduits, tiling, plastering, cabling, wiring and so on. So, now the constraint, the first constraint is laid the conduits before the tiles and plastering is done and there are two types of conduits, remember the electrical conduits and telecom conduits.

So, both the electrical conduits and the telecom conduits have to be done before the tiling and the plastering is done. Then it says lay the tiles and plaster the walls before you paint. So, the first, the second two steps must be done before the painting step. Painting has to be done before wiring and cabling has to be done and wiring has to be done before the fittings are installed in the electrical setup. So, this represents the constraints, this is a directed graph and this directed graph should not have cycles.

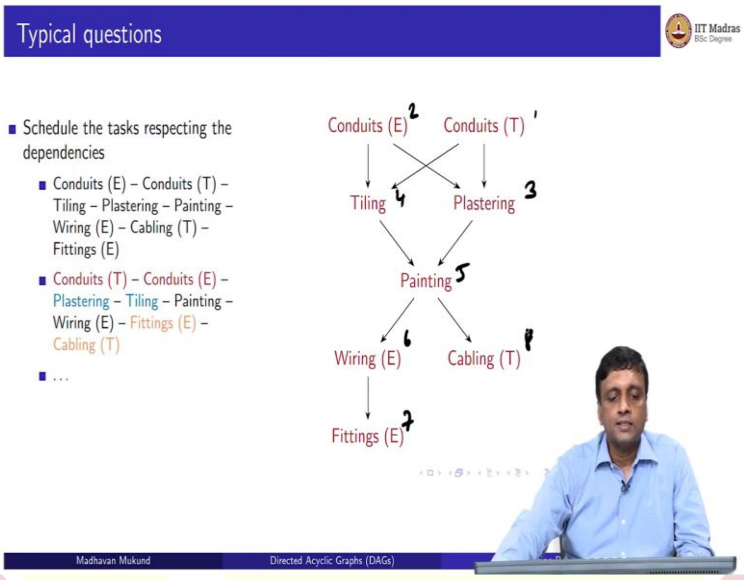
(Refer Slide Time: 03:06)



Typical questions

- Schedule the tasks respecting the dependencies
  - Conduits (E) – Conduits (T) – Tiling – Plastering – Painting – Wiring (E) – Cabling (T) – Fittings (E)

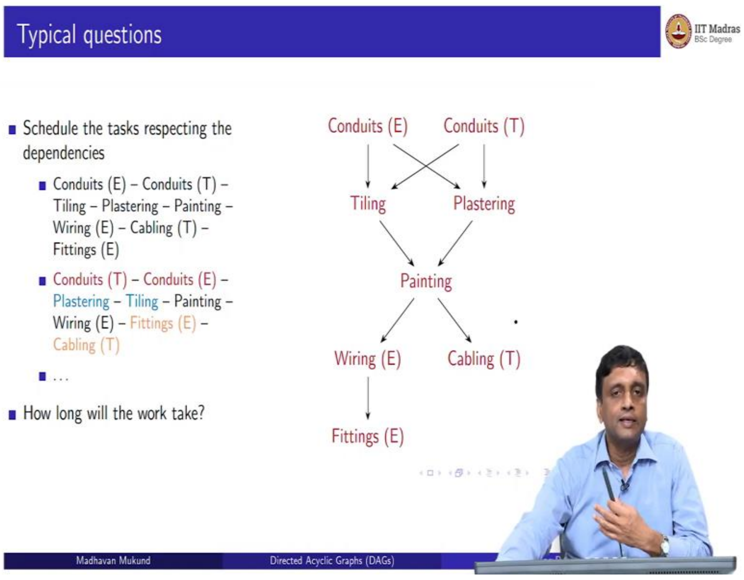
Madhavan Mukund Directed Acyclic Graphs (DAGs)



Typical questions

- Schedule the tasks respecting the dependencies
  - Conduits (E) – Conduits (T) – Tiling – Plastering – Painting – Wiring (E) – Cabling (T) – Fittings (E)
  - Conduits (T) – Conduits (E) – Plastering – Tiling – Painting – Wiring (E) – Fittings (E) – Cabling (T)
  - ...

Madhavan Mukund Directed Acyclic Graphs (DAGs)



Typical questions

- Schedule the tasks respecting the dependencies
  - Conduits (E) – Conduits (T) – Tiling – Plastering – Painting – Wiring (E) – Cabling (T) – Fittings (E)
  - Conduits (T) – Conduits (E) – Plastering – Tiling – Painting – Wiring (E) – Fittings (E) – Cabling (T)
  - ...
- How long will the work take?

Madhavan Mukund Directed Acyclic Graphs (DAGs)

So, if it had cycles, it would mean that I have to do a before B. But I also have to do B before A. So, that would be an unrealizable thing. So, we want to schedule these tasks so that these dependencies are respected and there may be many ways to do it. So, we can just do it like this. We can take the conduits first and then we can do the tiling and the plastering.

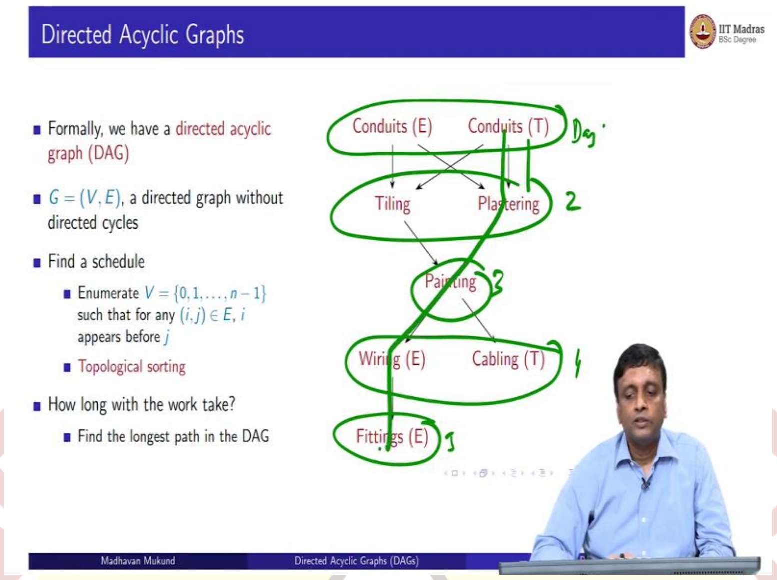
Then we can do the painting and then we can do the wiring and the cabling, then the fittings, so this would be one sequence in which you could do it. But if people are not available or there is some other optimization that we can do in terms of using manpower, we could also do some other things.

So, we could do conduits of the telecom before we do the conduits of the electrical thing and we can do plastering, before we do the tiling and then after we do the painting, we could do the electrical wiring and then finish it off, maybe the electrician is there, finish off the wiring and only then come to the cabling.

So, there are more than one way to process these tasks, while respecting the dependencies. So, that is one thing we want to know, what are the different ways in which we can execute these tasks, maintaining these dependencies and the other thing we might want to know is how long is the whole thing going to take, if I do this with as many people as possible, there is still some constraint that I cannot finish one job till I do the previous job and there may be some other constraint might be that I need the paint to dry.

So, if I paint the room today, maybe I will not be able to work in this room for another two days late. So, some of these tasks might have other things, plus we have the fact that we cannot violate this thing. So, if I have all my working staff available all the time, but I still have to do these and each of these has to be done on one or multiple days. What is the minimum number of days, so how long will it take is basically a minimization problem, what is the fastest I can do this?

(Refer Slide Time: 04:55)



Directed Acyclic Graphs

- Formally, we have a directed acyclic graph (DAG)
- $G = (V, E)$, a directed graph without directed cycles
- Find a schedule
  - Enumerate $V = \{0, 1, \ldots, n-1\}$ such that for any $(i, j) \in E$, i appears before j
  - Topological sorting
- How long with the work take?
  - Find the longest path in the DAG

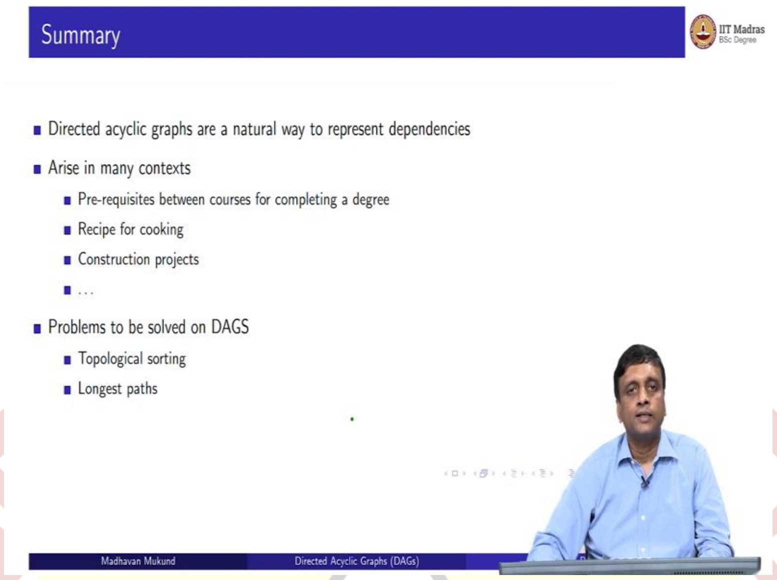
So, this is a directed graph without cycles, which is often called a Directed Acyclic Graph or a DAG. So, finding a schedule amounts to listing out these vertices in a sequence. This is the order in which I am going to process the work, you can think of it, in such a way that I never have an edge going from something which is later in the sequence to something earlier in the sequence.

So, I do not do conduits after plastering. So, in the sequence, all the edges must be respected by the sequence. So, if i appears before j, then there cannot be an edge from j to i, so this process of numerating the vertices in a consistent way with the constraints, constraints being the edges in my directed a cyclic graph. So, this is sometimes called topological sorting. So, I want to take a graph, which has many possible sequences, which are compatible and produce any one. This is called topological sorting.

The other thing, which is to ask how long I am forced to wait, is what is called the longest path. So, if I look at this, for instance you can see that if I start this and then I say, assume that everything can be done in a day, then on day 1 I can do this. Then on day 2, I can do these things and then day 3, I can do these things, day 4, I can do these things and only on day 5, I can do this.

So, I am guaranteed that I need 5 days to do this, there is no faster way no matter how I optimise it, there is no, it could take more than 5 days if I delay it, but I cannot do it in less than 5 days, because I have this longest path in my graph, which forces a dependency of 5 days.

(Refer Slide Time: 06:45)



Summary

- Directed acyclic graphs are a natural way to represent dependencies
- Arise in many contexts
  - Pre-requisites between courses for completing a degree
  - Recipe for cooking
  - Construction projects
  - ...
- Problems to be solved on DAGS
  - Topological sorting
  - Longest paths

So, we are going to look at how to calculate these things using our usual graph representations. So, to summarise, we have this notion of a directed acyclic graph. So, it looks like acyclic graphs are very simple and we should not need to worry about them. But actually, directed acyclic graphs are very important, because they very naturally capture this idea of tasks and dependencies.

So, anytime we have a constraint on doing things one before the other, this is what we need to analyse in order to understand how we can execute it without violating the dependencies and also how we how fast we can do it. So, topological sorting and longest path. So, these come in various forms.

For instance, you might have to satisfy some prerequisites before you complete a degree. So, what is the minimum number of semesters in which you can complete and graduate? Or you need to process some food for cooking. So, some things can be overlap, maybe you want to marinate something while you are chopping something.

So, maybe you have a lot of people in the kitchen who are helping out. But still, some steps must come before other steps. So, how fast can you actually prepare a dish? Or of course, in a construction project, the kind of thing that we said for preparing the room, also applies to a building as a whole, you have to lay the foundation, then you have to do some work on laying this slabs and then you have to do some work.

So, how fast can you actually put up a house or a building? So, all of these things have these natural constraints, which can be expressed in terms of DAGs and then these questions which arise, which is how do you process the tasks to satisfy the constraints and how fast can you do them.