Advanced Business Decision Support Systems

Professor Deepu Philip
Department of Industrial Engineering and Management Engineering
Indian Institute of Technology, Kanpur
Professor Amandeep Singh
Imagineering Laboratory
Dr. Prabal Pratap Singh
Indian Institute of Technology, Kanpur

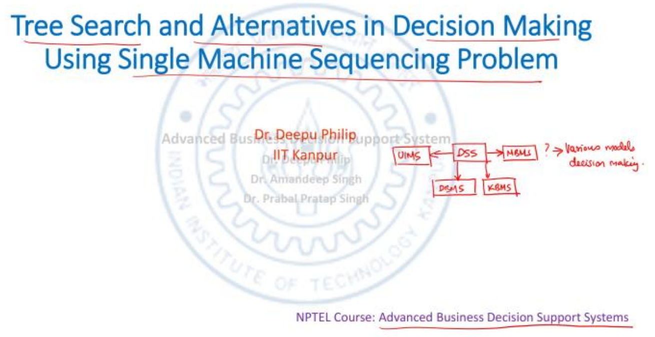
Lecture 09

Tree Search and Alternatives in Decision Making Using Single Machine Sequencing Problem (Part 1 of 4)

Good afternoon everyone, welcome to yet another lecture of Business Decision Support System, which is the advanced course of Web-Based Decision Support System that was plotted in the NPTEL MOOC's program. I am Dr. Deepu Philip from IIT Kanpur and along with me this course is Dr. Amandeep Singh Oberoi and Dr. Prabal Pratap Singh. And, we have been going through the process of Decision Support system and the type of business decisions and we have also discussed, what is the importance of the corporate decision making process and how the decision making process and the corporate decision why it is important to be covered all aspects of it.

And, we have also seen the simple aspects of Monte Carlo simulation and how Monte Carlo can be used for making decisions for where, you do not have clear values of a parameters etcetera. And then, in the last class, I mentioned that now we will get into what we call as the decision to the tree search or what we call as an Algorithmic Decision Making approach.

So, today we are looking into one of the decision models or a Decision Making Algorithm of tree search. So, today without any further delay let us get into the topic.



Tree Search and Alternatives in Decision Making
Using Single Machine Sequencing Problem

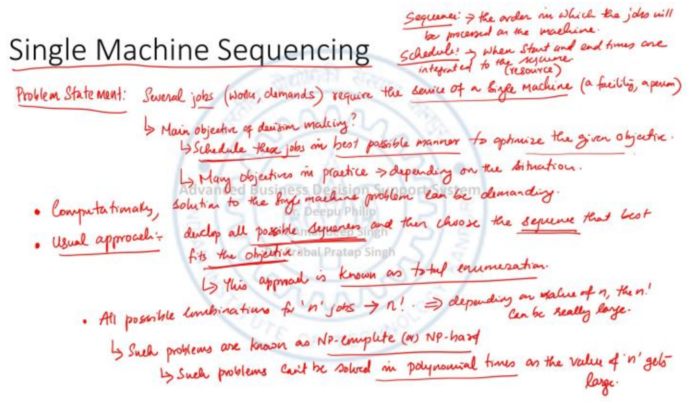
Dr. Deepu Philip
IIT Kanpur

NPTEL Course: Advanced Business Decision Support Systems

UIMS
DSS
MBMS
? → Various models decision making.
DBMS
KBMS

So, the tree search and alternatives, what are the alternatives of tree search in decision making and we are going to use a single machine sequencing problem as an example problem to demonstrate this. And, as I said earlier, this is part of Advanced Business Decision Support System course.

I really hope that you have already taken the previous course or you have known parts of the DSS and we also mentioned that in DSS, the major aspects are number one is UIMS (User Interface Management System). Then, we have what we call as DBMS (Database Management System) and then we have the third one is KBMS (Knowledge Base Management System) and then the last one is we talked about is MBMS (Model Based Management System). And, we have been seeing various models for decision making that is what we have been seeing as part of this course and some other things will be also caught out by my co-instructors.



Single Machine Sequencing

Sequence: → the order in which the jobs will be processed on the machine.
Schedule: → when start and end times are integrated to the sequence (resource)

Problem Statement: Several jobs (works, demands) require the service of a single machine (a facility, a person)
↳ Main objective of decision making?
↳ Schedule these jobs in best possible manner to optimize the given objective.
↳ Many objectives in practice → depending on the situation.

- Computationally, solution to the single machine problem can be demanding.
- Usual approach:- develop all possible sequences and then choose the sequence that best fits the objective.
  ↳ This approach is known as total enumeration.
- All possible combinations for 'n' jobs → $n!$ ⇒ depending on value of n, the n! can be really large.
  ↳ Such problems are known as NP-complete (or) NP-hard
  ↳ Such problems can't be solved in polynomial times as the value of 'n' gets large.

So, without any further delay, let us first talk about what is a Single Machine Sequencing problem? These are relatively problem to explain out of the concepts. So, if I put the problem statement what it is? In a simpler sense, it is several jobs, number of jobs. It can also be known as Works can be also known as Demands. Several jobs require the service of a single machine. So there is a single machine or a single resource.

So, it can be a machine or a resource you want to call it. A single machine can be a facility, a person, etc. So, several jobs or works or demands or etc. any of these falls into the jobs category. They require the service of a single machine or a single resource.

It can be a facility, it can be a person, it can be drilling machine, can be a milling machine, etc. And, within this, the first thing is, the main objective is Decision Making. What is the main objective of decision making? Schedule these jobs in best possible manner to optimize some given objective. So, you schedule these jobs in a sense with the start time and end time and order in which, it will be processed in the best possible manner. So that you can for the given objective can be optimized.

The other part of it is, there are many objectives in practice not just one. But that is depending on the situation. Whatever is the situation depending on that is the objective that is usable. Now, other point you should also understand that computationally solution to single machine is demanding. Solution to the single machine problem can be demanding.

You may require a lot of computing resources to solve this problem. And, the usual approach, what do we do usually? The usual approach is develop all possible sequences and then choose the sequence that fits the best objective. So, there are two words that we have seen today quickly. One is Schedule and another one is Sequence. So, we will discuss this in a minute, but first, the usual approach is develop all possible sequences and then choose the sequence that best fits the objective.

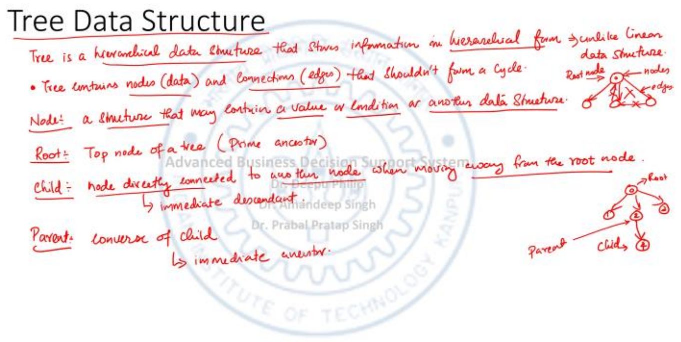
So you develop all possible sequence. So this approach is known as Total Enumeration. Get all possible sequences and from there, pick the sequence that best fits the objective. Now, you have discussed two words Sequence and Schedule. What are those? They are related to one another.

Sequence is the order in which the jobs will be processed on the machine. It will just order the precedence which will be the first job, which will be the third job, which will be the seventh job, etc. Schedule is when start and end times are integrated to the sequence. So sequence is just the order, there is no start and end time whereas, in a schedule it, is a sequence with a start and end time. So, the job 1 will start processing at 9 and finish at 907, job 2 will start at 908 and then finish at some other number, etc. that is called as the Schedule. So Schedule and Sequence are this way related.

Then, we mentioned about the Total Enumeration. Total enumeration is where all possible sequences are first developed and after that you choose a sequence that best fit the objective. So, all possible combination for n jobs, let us say there are n jobs in the system, you need 1, 2, 3, 4 etc. up to n. Then, if that is the case, then all possible combination of n jobs will be n factorial. So, if it is 10 jobs, it will be 10 factorial, depending on value of n, the n factorial can be really large, can go like crazy.

So, such kind of problems are known as NP complete or NP hard. So, this you might have heard this term NP complete or NP hard or etc. So, in a simpler sense for us, what does means is, such problems cannot be solved or you are unable to solve those problems in polynomial times as the value of n gets large.

So, you cannot solve in polynomial times, as the value of n gets large, the time starts growing exponentially not polynomial. So, that is the issue with such kind of problems. So, as we said the aim here is to develop all possible sequences first and then choose a sequence that best fits the objective and we told that is called Total Enumeration. So, we are going to look into that and then go from there that is our plan for today.



Tree Data Structure

Tree is a hierarchical data structure that stores information in hierarchical form → unlike linear data structure.

• Tree contains nodes (data) and connections (edges) that shouldn't form a cycle.

Node: a structure that may contain a value or condition or another data structure.

Root: Top node of a tree (Prime ancestor)

Child: node directly connected to another node when moving away from the root node.
↳ immediate descendant.

Parent: converse of child
↳ immediate ancestor.

So, now, to do this, we have to learn also about Tree Data Structure. What is a Tree Data Structure? So, in simplest ways, tree is a Hierarchical Data Structure. So, hierarchical data structure that stores information in hierarchical form is unlike linear data structures like array or something data structure. So, this is hierarchical data structure that stores information in the hierarchical form that is what a tree thing is. So, important things are tree contains nodes. Nodes usually you can think about as data and connections you can think about it as edges.

Edges is a graph kind of a term that we use that should not form a cycle. So, it cannot have a cycle. The people draw tree in a multiple fashion but the idea is that a tree contains nodes and edges. So if you represent a tree something like this, the easiest way to represent a tree is like this. So, these are nodes and these are edges and the idea is that you know, do not make things like this.

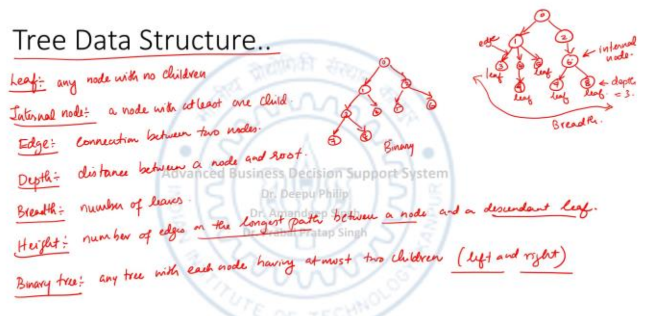
So, that you do not create a cycle in that fashion. So, now let us define the term node. For this, I think you may all have studied this but I am just doing this for the purpose of this course . So, we call node here in this course for the Decision Support System a structure that may contain a structure, a value or condition or another data structure. So, a node is a structure, it is a way to store data that may contain a value can have a value, a condition or another data structure.

So, internally, we remember that we are using computers to do it. So, it is implemented in that way node. Then, there is a word that you come across is called the Root. Root is the top node of a tree for this is the 0 or the root node. Top node of a tree is called also known as Prime Ancestor. That is a top node.

Then child is another term that you need to think about. Child is the node directly connected to another node. A node that is directly connected to another node when moving away from the root node. So, the word for this is called as Immediate Descendant. So, a node that is directly connected to another node, when moving away from the root node.

So, if you draw a diagram like this is the root node, having a node like this 3 and another node. So, this can be said as a child. So, this is 1, 2, 3 and 4. So, the 4 is the child of 2, the immediate descendant of 2. Then, let us call about what is parent.

It is the converse or reverse of child . So, that is the Immediate Ancestor. So, if I use this diagram, this is our parent and child is for the 2 is the parent not 4 is the child.



Tree Data Structure..

Leaf:- any node with no children

Internal node:- a node with atleast one child.

Edge:- connection between two nodes.

Depth:- distance between a node and root.

Breadth:- number of leaves.

Height:- number of edges on the longest path between a node and a descendant leaf.

Binary tree:- any tree with each node having atmost two children (left and right)

Binary
edge
internal node.
leaf
leaf
leaf
leaf
leaf
depth = 3.
Breadth.

Now, with this, let us continue a little bit more on the Tree Data Structure and more terms and we will be using these terms when we are discussing it. So, it is better that we understand this clearly.

So, leaf is the another word that we need to learn. It is any node with no children . So, an example of it is, you have a node 0 and you have 2 nodes like this 1 and 2 and from here you have 3 nodes 3, 4 and 5 and that is one node 6 and let us say 6 has 7 and 8 and 4 has 9. So, then, this is a leaf, leaf, leaf and leaf. So, the nodes that does not have any children, there is no immediate descendant, then that node is called as a leaf node.

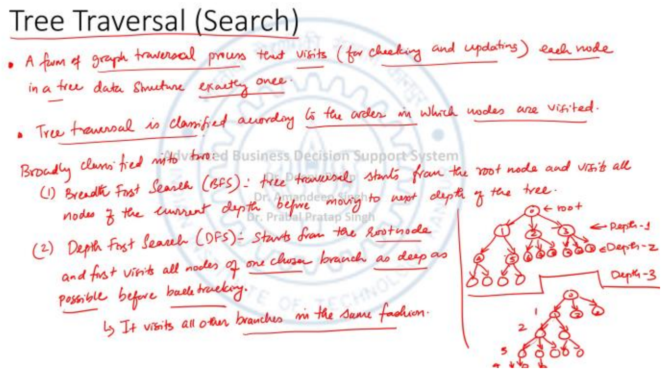
Then, internal node: a node with at least one child. There is at least one child, then that is called as a internal node. So, in this case, you can say that 6, 1, 2 etc. are internal nodes. Then, next word we are going to deal with this is Edge.

What is an edge? So it is a connection between 2 nodes. So 1, 2, 3 this is an edge, it is connected by an edge. Then, the depth is, another word that you need to realize is the distance between a node and root. So, if you try to find out the depth of 8, it is from here level 1, level 2, level 3.

So the depth here is for 8 is equal to 3 . So, that is the depth of 3 from the root. Breadth and depth are important because our algorithm also will be using these terms. Breadth is the number of leaves, how broad the tree is determined by the number of leaves. So, this is the breadth the number of leaves determines the breadth. Then, height, number of edges or connections on the longest path between a node and a descendant leaf.

So, for any given node, you can take that and then go through the node, take the longest path and find where that leaf is available. That descendant leaf is there from a node and the descendant leaf, whatever is the longest path is called as the Height of the tree. Then, the last term we need to do is Binary tree. You will hear this term many other times. So, any tree with each node having at most 2 children, cannot exceed beyond 2 children, that is it. So, it is called as the left and right nodes.

So, if you take a tree and then you have like this 0, 1, 2, 3, 4, 5, 6, this will have 7, 8 like this. This will become a binary tree at most 2 nodes. So, there is the left and the right left node and the right node.



Tree Traversal (Search)

- A form of graph traversal process that visits (for checking and updating) each node in a tree data structure exactly once.
- Tree traversal is classified according to the order in which nodes are visited.

Broadly classified into two:
(1) Breadth First Search (BFS): tree traversal starts from the root node and visits all nodes of the current depth before moving to next depth of the tree.
(2) Depth First Search (DFS): Starts from the root node and first visits all nodes of one chosen branch as deep as possible before backtracking.
↳ It visits all other branches in the same fashion.

So, now let us take a look into the algorithm how do we do the Tree Traversal or what we call as the Search Tree. Tree Traversal is also known as Tree Search. So how do we define this? It is a form of graph traverse or traversal process that visits. Why do we need to visit for checking and updating? It visits for checking and updating each node in a tree.

So, it is a form of graph traversal process and it visits each node in a tree and the purpose of the visit is for checking and updating. So, each node in a tree data structure exactly once, you will only do once, you will visit the node exactly once . That kind of a traversal process is what we call as Tree Search or Tree Traversal.

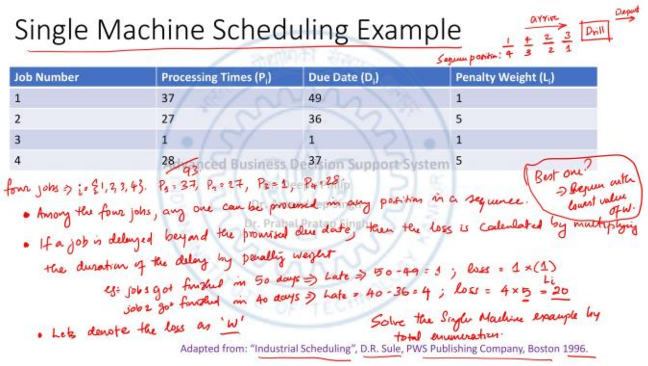
Then, tree traversal is classified according to the order in which nodes are visited. So, we classify the way the order in which, you visit the nodes that classifies the tree traversal algorithm. So, broadly classified into two. Number one, it is called as the Breadth First Search (BFS). Then, it starts the Traversal Tree from the root node and visits all nodes of the current depth before moving to next depth of the tree.

So, the way to think about it is, this is the root node. Let us say it has 4 nodes or 3 nodes 1, 2, 3. It first visits all these 3, so this is the depth. Then, it will go to the nodes below it, all of them like this. So, this is depth 2, 4, 5, 6, 7, 8, 9, 10, 11 then, it will visit the next depth like this.

So, this is called as the breadth first where you finish all nodes in a one particular depth and then, you move to the next depth, that is the Idea. The second one is called as the Depth First Search, we call it as DFS. So, it starts the Tree Traversal from the root node again. All begins at the root node and first visits all nodes of one chosen branch as deep as possible before backtracking. So, what it does is, it starts from the root node and then first visits all nodes of a chosen branch as deep as possible before backtracking.

So, what happens here is, an example of that, let us take, this is the root node and there are as we said 3. So, the 3 is expanded, then we take this node in which this expanded like this further. Then, it goes to the next one and let us say, this is where it stops. So, first, it will go through in this fashion and reach this depth. So, this is depth 1, depth 2, depth 3 and depth 4, then it will backtrack.

There is no other nodes, here it will backtrack, it will go to this node, go backtrack come here there is nothing then, come back backtrack, here it will expand like this. So, that is the process of depth first search. So, remember that it visits all other branches in the same fashion. As I just explained to you, in the depth first where it goes down as deep as possible and then backtracks and goes through this. This is the approach that it follows in the other branches of this.



Single Machine Scheduling Example

$\xrightarrow{\text{arrive}}$ $\boxed{\text{Drill}}$ $\xrightarrow{\text{Depart}}$

Sequence position: $\frac{1}{4}$ $\frac{4}{3}$ $\frac{2}{2}$ $\frac{3}{1}$

Job Number	Processing Times (P_j)	Due Date (D_j)	Penalty Weight (L_j)
1	37	49	1
2	27	36	5
3	1	1	1
4	28	37	5

93

four jobs ⇒ $i = \{1,2,3,4\}$. $P_1 = 37$, $P_2 = 27$, $P_3 = 1$, $P_4 = 28$.

- Among the four jobs, any one can be processed in any position in a sequence.
- If a job is delayed beyond the promised due date, then the loss is calculated by multiplying the duration of the delay by penalty weight

eg: job 1 got finished in 50 days ⇒ Late ⇒ $50 - 49 = 1$; loss $= 1 \times (1)$

job 2 got finished in 40 days ⇒ Late $= 40 - 36 = 4$; loss $= 4 \times \underset{\_}{\overset{L_i}{5}} = \underline{20}$

- Lets denote the loss as 'W'

Best one? ⇒ Sequence with lowest value of W.

Solve the Single Machine example by total enumeration.

Adapted from: "Industrial Scheduling", D.R. Sule, PWS Publishing Company, Boston 1996.

Now, we have seen the Tree Scheduling and Tree Search as such so not scheduling. So, now, let us talk about the Single Machine Scheduling Example. So, this is an example taken from the industrial scheduling textbook by D.R. Sule, PWS Publishing Company, 1996 and say example that is small enough for us to work out in the class and we can understand.

So, there are 4 jobs, where j is equal to 1, 2, 3, 4 and each not here, we are using the index. So, let us put use index i, it is job. So, then the processing times Pi sub i. So, the P1 means, processing time of job 1 is 37, P2 is processing time of job 2 is 27. P3 is the processing time of job 3 is 1, P4 processing time of job 4 is 28.

So, that is the total processing time of all the jobs. So, then, if we have to process all these jobs without any delay in just 1, 2, 3, 4 then, the sum total of this. So, that is 9 plus 7 is 16, so this is about 93. The total duration, time required for doing all of this is 93 as such. The due date means, the date at which you have promised the customer that you will deliver the job. For job 1 it is 49, job 2 is 36, job 3 is 1 and job 4 is 37 and there is a penalty weight that is given.

Penalty weight means, if the job is delayed by x number of units, then the penalty weight is multiplied with that to find out what will be the penalty that you will be paying to the customer, The other things you need to remember is among the 4 jobs, anyone can be processed in any position in a sequence.

So, what we have here is, let us say, we have a single machine, let us say it is a drilling machine . So, the jobs arrive into the system like this and then, they come in and then they get drilled and depart, that is a single machine . So, you have sequence position, your 4 jobs means your 4 sequence position 1, 2, 3, 4, like this. So, 1 means, the one that will be processed first, 2 is the second, third, fourth, etc.

So, you can do like this 3, 2, 4, 1, etc. That means, job 3 will be processed first followed by 2 and then 4 and 1, etc. That is the idea . So, these are the sequence positions. That is what we want below. Then, if a job is delayed beyond the promised due date, the loss is obtained by loss is calculated by multiplying the duration of the delay by penalty weight.

So, for example, if job 1 got finished in 50 days - lateness late equal to 50 – 49 that is 1. The loss is equal to 1, which is a delay multiplied by 1, which is this is the Li the penalty weight. If job 2 got finished in 40 days, that is the case the lateness is calculated by 40 – 36 that is 4. So, the loss is calculated as 4 times 5 is 20. So if job 1 got laid by 4 because the penalty weight is 1, it will only be the loss will be 4, but if job 2 got delayed by 4, then the penalty weight will be, total loss will be, 20 because of the penalty weight of 5 .

So, let us denote the loss as W. We use the letter W to denote the loss in the system . So, I hope you guys have understood what we have just discussed here. So, we talked about what is a Single Machine Sequencing, we talked about what is a Tree Data Structure and the roots and the nodes and child parents and the leaf internal node the depth, breadth, height, etc. All those aspects of tree we have discussed. And then, what is Tree Traversal, how to the traversal processes visits each node in a tree exactly once for checking and updating and there it is classified into two, the Breadth First and the Depth First.

And, what Breadth First does, what Depth First does, we discussed that and then we just talked about what is a Single Machine Scheduling example and what it looks like and once that it is done and what is the loss and how the loss is also calculated and loss is denoted by W. Now, with this our job is to solve this problem, the Single Machine Scheduling example problem by total enumeration. So, what we will do is, solve the Single Machine example by total enumeration. What is total enumeration? So, we already discussed in total enumeration is that we just here we visited develop all possible sequences and then choose a sequence that is best fit the objective .

So, this approach is known as the total enumeration. So, we are going to develop all possible options, all possible alternatives and choose the best one. The best one is the one, that has the lowest width. So, what is the best one here? What is that one? Sequence with lowest value of W. The lowest value of W will be our the best sequence in this.

So, that is what we are going to do as part of today's example. So, we will start enumeration in the next part. So, I really hope you guys will follow this enumeration with pen and paper because this is not very easy to understand. So, I will work step-by-step, so it will be easy for you to understand.

So, we will take a small break and we will start the total enumeration in the next session.

Thank you.