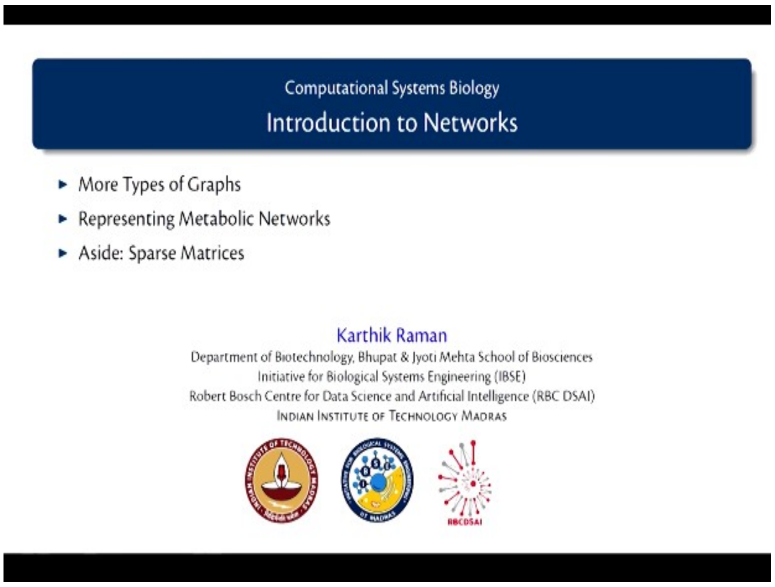
Computational Systems Biology

Karthik Raman
Department of Biotechnology
Indian Institute of Technology – Madras

Lecture - 14
Introduction to Networks

In this video, we will continue with our introduction to networks and look at more types of graphs like hypercube graphs and so on.

(Refer Slide Time: 00:14)

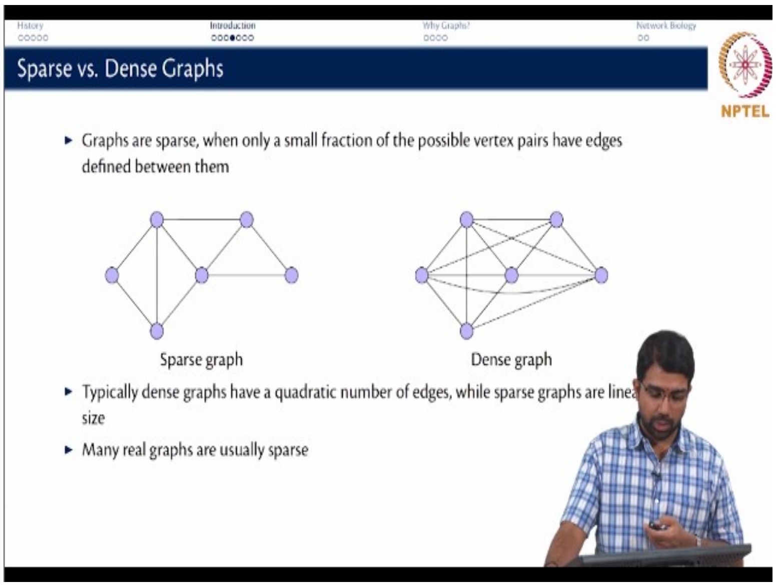

Computational Systems Biology
Introduction to Networks

- More Types of Graphs
- Representing Metabolic Networks
- Aside: Sparse Matrices

Karthik Raman
Department of Biotechnology, Bhupat & Jyoti Mehta School of Biosciences
Initiative for Biological Systems Engineering (IBSE)
Robert Bosch Centre for Data Science and Artificial Intelligence (RBC DSAI)
Indian Institute of Technology Madras

And we will also start looking at important graphs in biology namely metabolic networks and we will have a brief aside where we discuss about storing and representing sparse matrices on a computer.

(Refer Slide Time: 00:32)

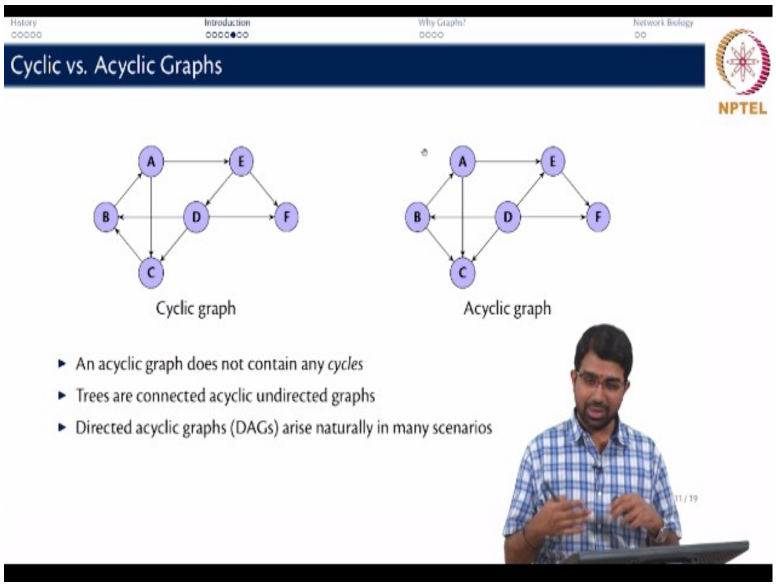


Sparse vs. Dense Graphs

- Graphs are sparse, when only a small fraction of the possible vertex pairs have edges defined between them
- Typically dense graphs have a quadratic number of edges, while sparse graphs are linea size
- Many real graphs are usually sparse

So, your graphs could be dense or sparse and you will typically find that most biological networks are quite sparse, in the sense, there are very few interactions for every node, relatively. If you have thousands of nodes and a node might actually interact only with a hundred nodes at max, usually with three or four, but occasionally with hundred, but never more than ten percent of the nodes or something like that and this has implications for how you do the math with these kinds of networks and so on.

(Refer Slide Time: 01:02)



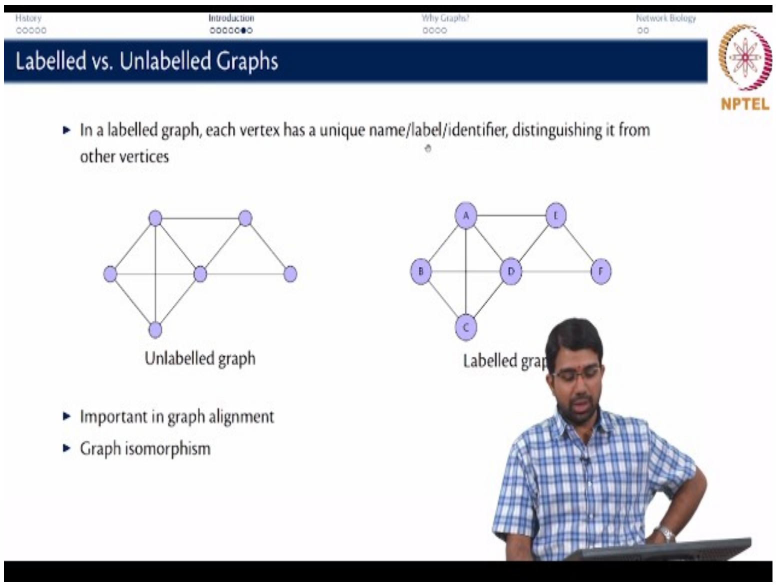
Cyclic vs. Acyclic Graphs

- An acyclic graph does not contain any *cycles*
- Trees are connected acyclic undirected graphs
- Directed acyclic graphs (DAGs) arise naturally in many scenarios

And then you have cyclic graphs. You can imagine that metabolic networks will be cyclic right. You have several cycles like the tricarboxylic acid cycle and so on. So, you will see that there is a path from a node to itself. So, you start at A on this graph, you go from A, you go to C, you go to B, you can come back to A whereas, in this graph, you find that there is no such path from A to itself.

So, trees are essentially connected acyclic, undirected graphs. They can also occur in metabolism. Whenever you remove cycles, you will find that there are directed acyclic graphs that arise and they also arise in scheduling and so on. Let us not worry about that.

(Refer Slide Time: 01:48)



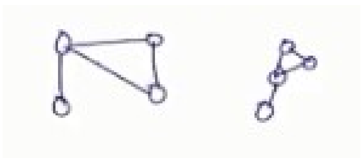
Labelled vs. Unlabelled Graphs

- In a labelled graph, each vertex has a unique name/label/identifier, distinguishing it from other vertices

- Important in graph alignment
- Graph isomorphism

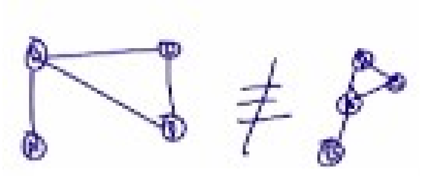
Another useful concept is the concept of labelling a graph. So, a graph is unlabelled versus labelled.

(Refer Slide Time: 02:03)



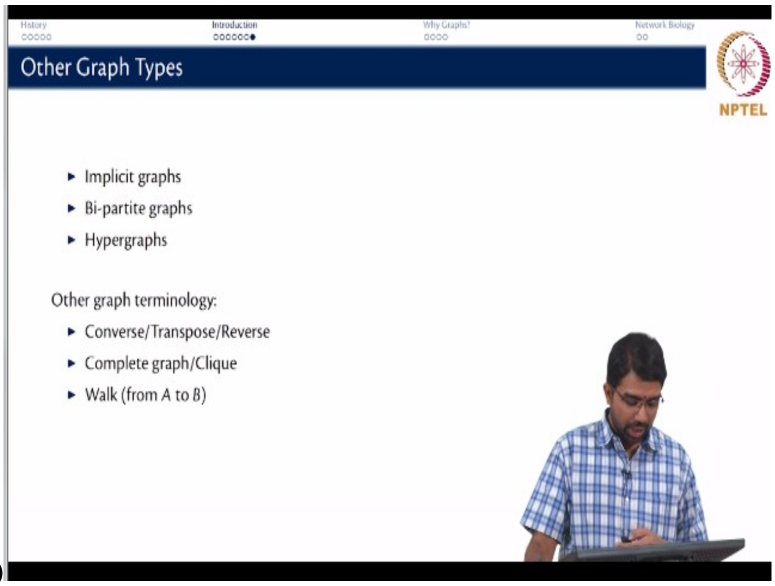
Unlabelled and isomorphic

There 2 graphs are the same right, if I didn’t label them. If I label them, the graphs are no longer isomorphic.



Labelled and not isomorphic

So, till you label these graphs they were equal. So, these have very interesting applications in aligning proteins, aligning protein interaction networks themselves and so on.



Other Graph Types

- Implicit graphs
- Bi-partite graphs
- Hypergraphs

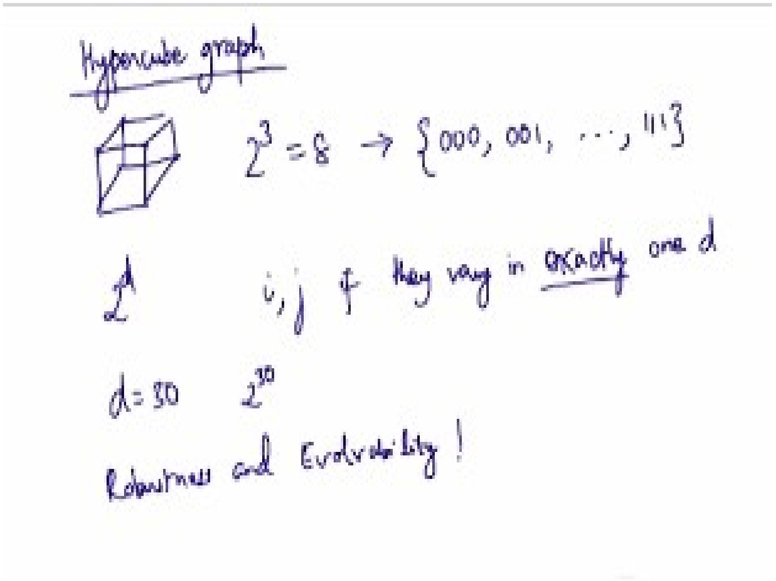
Other graph terminology:

- Converse/Transpose/Reverse
- Complete graph/Clique
- Walk (from A to B)

(Refer Slide Time: 02:53)

And then you have this notion of an implicit graph. So, these are graphs where you can describe what node connects to what node, but you never spell out the entire structure of the graph or something like that.

(Refer Slide Time: 03:14)



Hypercube graph

$2^3 = 8 \rightarrow \{000, 001, \ldots, 111\}$

2^d i, j if they vary in exactly one d

$d = 30$ 2^{30}

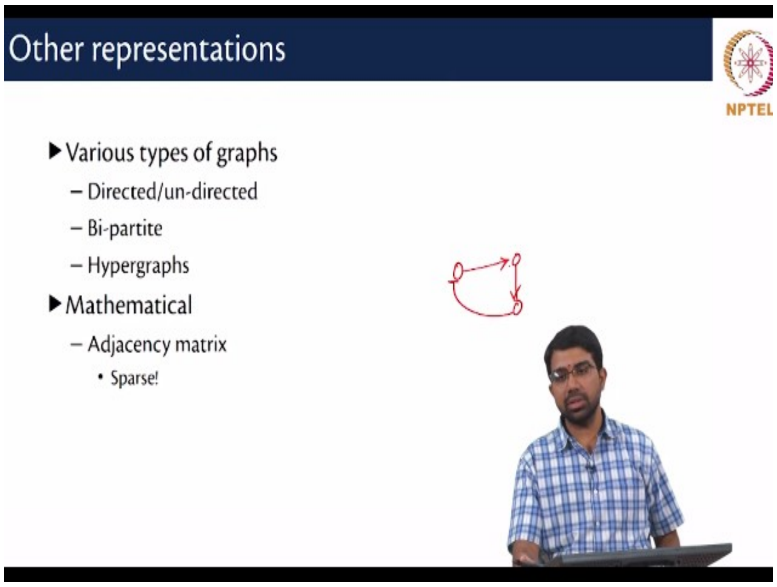
Robustness and Evolvability!

So, a classic example of an implicit graph would be something like a hypercube graph where basically this is a 3-dimensional hypercube graph, right. How many nodes are there? Basically 000, 001,111 and in a d-dimensional hypercube you will have 2^d nodes right and d dimensions and you will have edges between i and j, if they vary in exactly one dimension.

So, you may never be able to store this graph on a computer. If d is 30, 2^{30} is a very, very large number right, it is like a billion roughly. So, you would not be able to store it on a computer, but you can run several graph algorithms, right and in fact hypercube graphs have implications for what we will study at the very end of this course, the advanced topics that I mentioned earlier on.

So, hypercube kind of graphs are very prevalent there. Another very useful concept is that of bipartite graphs and so this brings us back to how do you represent metabolic networks.

(Refer Slide Time: 05:09)

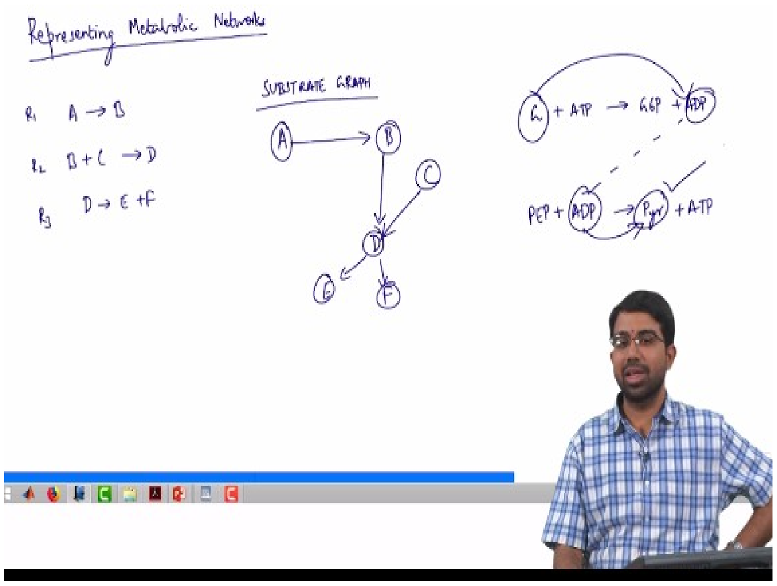


Other representations

- Various types of graphs
  - Directed/un-directed
  - Bi-partite
  - Hypergraphs
- Mathematical
  - Adjacency matrix
    - Sparse!

How do you represent a metabolic network?

(Refer Slide Time: 05:20)

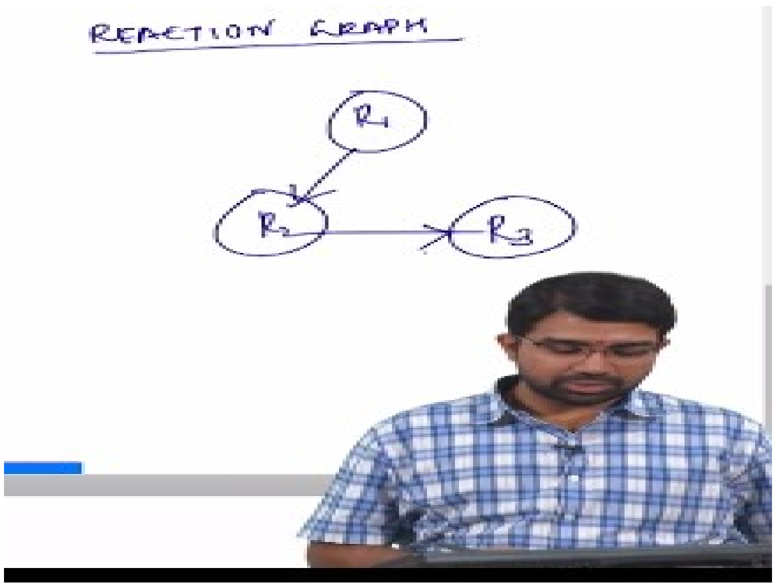


So, this is your network. How would you represent this as a graph? There are various ways to do it and depending upon your choice, you will end up with a different kind of representation. So, one way to represent this would be something known as a substrate graph. As the name suggests, in a substrate graph the nodes are all substrates, in this case it will be A, B, C, D, E, F and the edges represent those substrates which can be converted to one another.

So, A, B, D. Do you see a limitation with this kind of a representation? You don't know, right? So, this again brings us to a limitation of modelling. So how do you choose a representation? Going back to how would you choose a model? Depending on what you want. So, if I want to figure out what is the shortest path to produce F from A or let's just for a moment think about glycolysis. I will just write 2 reactions, glucose+ATP, is that right? That would be the first reaction of glycolysis. What is the last reaction of glycolysis? Phosphoenolpyruvate+, I have a problem here. I now have a two-step glycolysis. Convert glucose to ADP, convert ADP to pyruvate. Wow! I have glycolysis in two steps. This doesn't make any thermodynamic sense, this doesn't make any biological sense, but it is potentially a conclusion you would draw from the substrate graph.

One way to get around this is just get rid of ATP, right. Because you will find that these nodes have very high degree in the graph, you just get rid of this then you are stuck. Then you have to find a way to convert glucose to PEP and PEP to pyruvate. You will find that this will help you trace out something that is most likely your real glycolysis, but naturally you want to look for other ways to represent these graphs. Another way to represent this is what known as a reaction graph.

(Refer Slide Time: 09:02)

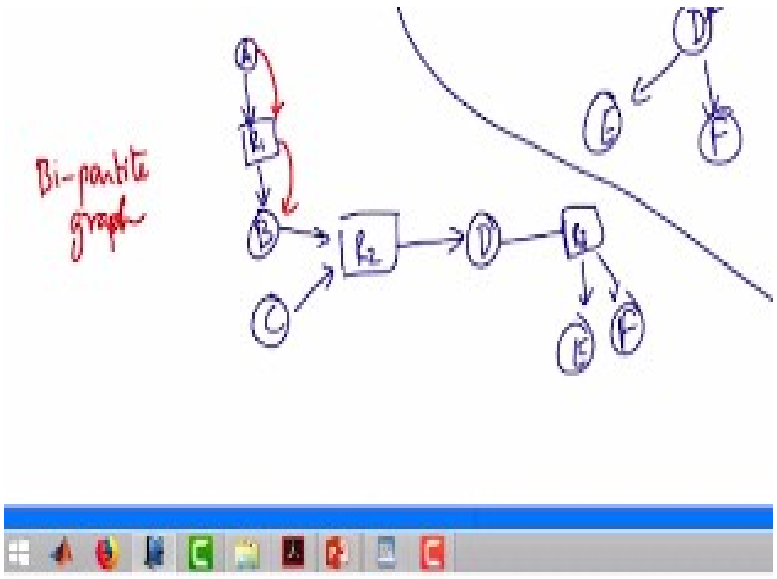


As the name suggests, your nodes are R1, R2, R3. When do you connect them? If they can operate consecutively, right. So, after R1, you can have R2 and after R2 you can have R3.

There is a notion of direction again. But again you see, very limited in the information. So, since we are not happy with either of these let's try to look at how a textbook picture would normally look.

How would your textbook picture look? It will be basically something like A, you will not have R1 in the box, you would actually have the first enzyme in the box, right. And then you will have B, C. R2 gives D. And D, R3, E, F.

(Refer Slide Time: 10:15)



Do you notice something peculiar about this graph? You will note that there are no links between circles or no links between squares right. So, you find that all links are between circles and rectangles or rectangles and circles. So, this is called a bi-partite graph. Because you can partition the nodes into two, such that only one bunch of nodes connect with the other bunch of nodes, no connection between the nodes.

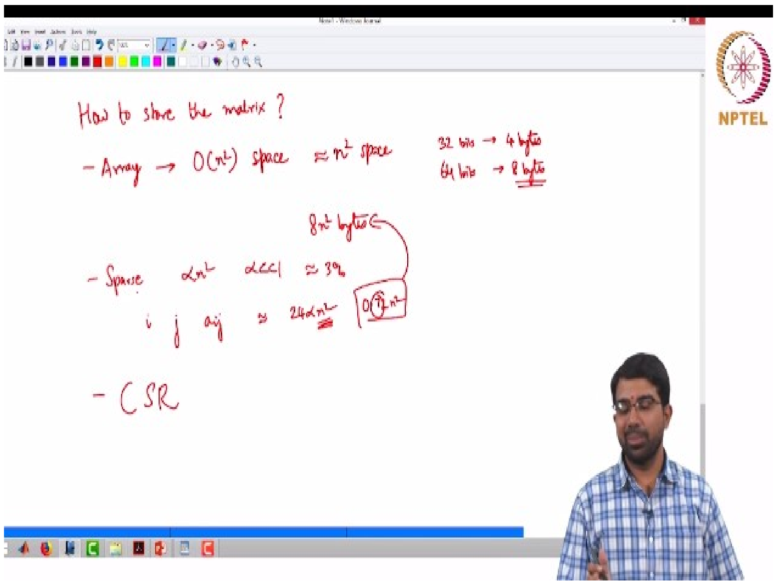
This also has an aside, quick aside, how do you represent these graphs? You finally need to represent them on a computer, we look at it obviously much more closely in the next lab session where we will have an intro to MATLAB, but usually use something known as an adjacency matrix. What is an adjacency matrix? It is a matrix size of number of vertices in the graph.

aij equal to 1 if there is an edge between i and j. Immediately you can think of multiple things. For a weighted graph the adjacency matrix will carry numbers, weights. In an unweighted graph, it will just be binary, 0 and 1. In a directed graph, the adjacency matrix will be asymmetric, whereas it will be symmetric in an undirected graph or you will essentially only store the upper triangular matrix or something like that.

How will it look in a bi-partite graph? If I said these are my metabolites and these are my reactions, the circles and your rectangles, how will the adjacency matrix look like? They will be non-block diagonal, right? This will be 0, this will be 0, no intra-partition links, right. You will have one's only somewhere here, right. So, you will have a, it will be a block matrix, obviously this is after you reorder the rows.

If you interleave the reactions and reaction and metabolite nodes, you wouldn't have this but if you reorder them by the partitions like the R partition and the M partition you would observe this kind of a matrix. And one more important thing you need to think about is, how to store the matrix?

(Refer Slide Time: 13:31)



How to store the matrix?

- Array $\rightarrow$ $O(n^2)$ space $\approx n^2$ space

32 bits $\rightarrow$ 4 bytes

64 bits $\rightarrow$ $\underline{\underline{8 \text{ bytes}}}$

$8n^2$ bytes

- Sparse αn^2 $\alpha << 1$ $\approx 3\%$

i j a_{ij} $\approx$ $\underline{\underline{24\alpha n^2}}$ $\boxed{O(\alpha n^2)}$

- CSR

You typically use an array. What is the problem with an array? It takes up or if you are not familiar with big O notation, it basically takes approximately n^2 space. So, in fact if you are looking at storing doubles let's say how much space is taken up by a double in computer memory? 16, 32, 64? So, it depends really on the precision right. So, most machines will have at least 32 bits or 4 bytes.

This is in fact for a float. For a double, it is usually 64 bits and 8 bytes. So, you are going to store 8 bytes. So, $8n^2$ bytes is what you need. But what did we say already, we said that most real graphs or most biological graphs are actually sparse. So, which means maybe I don't have, let's says the number of nonzeros I have is some αn^2 where $\alpha<<1$.

Maybe I have 3% nonzero entries, right. So, it is wasteful to store the entire matrix in memory, I mean as you know a full matrix with basically having 97% entries are just being filled up with 0s. So, is there a better way to store these matrices? Is there a better, you can just store that, that would be a classic way? You just store i, j, aij, how much space does this take? okay, into 8 so that will be $24\alpha n^2$.

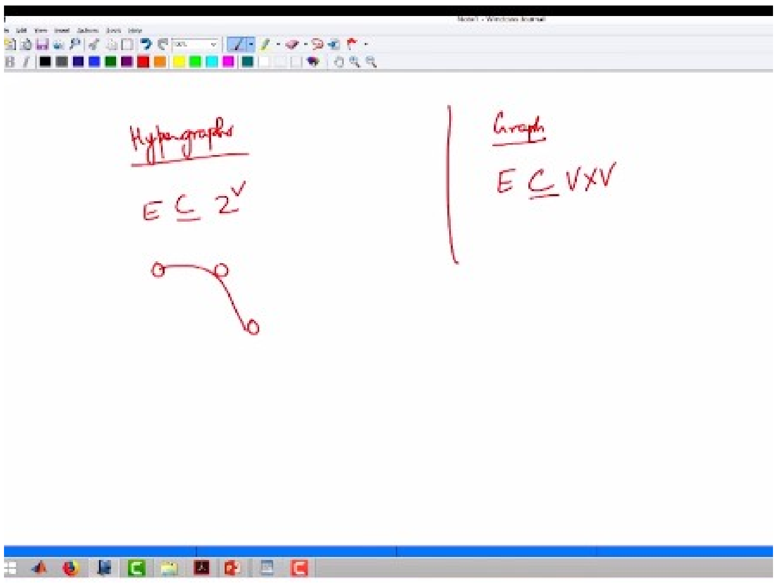
And given α is, we said 0.03 right, so it will be 0.72, right. So that is much, much smaller number compared to this. So, in practise you will obtain a lot of savings when you store it like this, but this is also a naïve method, 0.72 is much smaller than 8 right. So, you made heavy savings. There are other methods which I want you to go back and read about. There is something known as CSR, it stands for compressed sparse row format.

There are many other formats to store these matrices and the advantage is how fast is it to compute A^T, how fast is it to compute A^TX. These are things you would normally learn in the numerical methods course. So, how sparsity can be exploited in computations. CSR is, see none of this is, so n^2 is actually you know misnomer here, you should just say it is linear in the number of nonzeros, right.

So, this is already linear in the number of nonzeros. This will be again linear in the number of nonzeros but even smaller. You only store the runs basically where do my nonzero start and how long is my run of nonzeros and what are the positions that they occupy and so on. So, these are some very important concepts because you will, the moment you start working with biological problems, you will see that the size of the data, the size of the matrices become very challenging.

So, you may not even be able to load a data set unless you try to cast this as a sparse matrix. So, these are the challenges that you have to normally worry about. And then you have hypergraphs which are nicer generalizations.

(Refer Slide Time: 17:52)



So, in hypergraphs you will see that, so in a graph, is that right, whereas in hypergraph you can have more than one node connected by an edge, more than 2 nodes connected by an edge. So, you have A, B, C. You have a hyper edge that connects these 3 nodes, which means that there is no way to just have a path, there is no notion of a path between A and B now. There is a path called A, B, C.

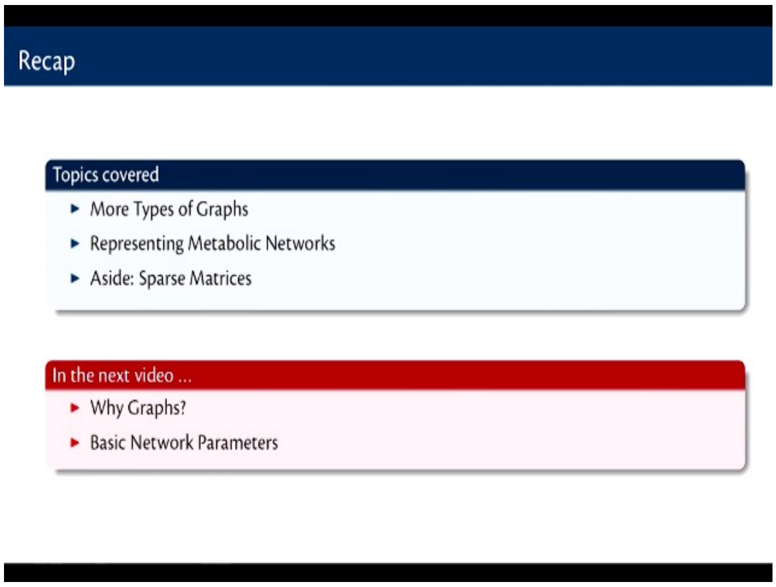
And can you see where this would be useful? Already from what we have seen today? Metabolic networks, right? Metabolic networks are naturally represented as hypergraphs because you need A and B, you need glucose and ATP to produce glucose 6 phosphate. You can't just convert glucose to G6P. So, do read more about hypergraphs, they are quite important.

And then we have the notion of converse of a graph or a transpose of a graph and so on. And basically, you invert the edges. If A is connected to B, you disconnect it and whatever other edges don't exist in the graph you add it. So, essentially the sparsity will also invert, the graph is 3% sparse. The inverse will be 97% dense, not sparse right. And you can have a complete graph or a clique where every node is connected to every other node.

We will look at these in a little more detail in the next lecture. You also have a notion of a walk from A to B right. So, a walk is essentially node-edge-node-edge-node. So, A take the edge AB, B take the edge BC, reach C. So, it traverses nodes and edges so this is very important when you look at graph algorithms. We will not focus on graph algorithms in this course, but we will instead study how you know various network parameters, various network topologies are used to understand and infer about graphs.

I think I will stop here for today.

(Refer Slide Time: 20:03)



Recap

Topics covered

- More Types of Graphs
- Representing Metabolic Networks
- Aside: Sparse Matrices

In the next video ...

- Why Graphs?
- Basic Network Parameters

So, today we looked at a few more types of graphs and how one represents metabolic networks and what are the advantages and disadvantages or you know the kind of interpretations one can make with different types of metabolic network representations. And we also overviewed some concepts of storing sparse matrices on a computer. In the next video, we will look at some more motivation for why we use graphs and the fundamentals of network biology beginning with various types of network parameters.