Algorithms for Protein Modelling and Engineering

Professor Pralay Mitra
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

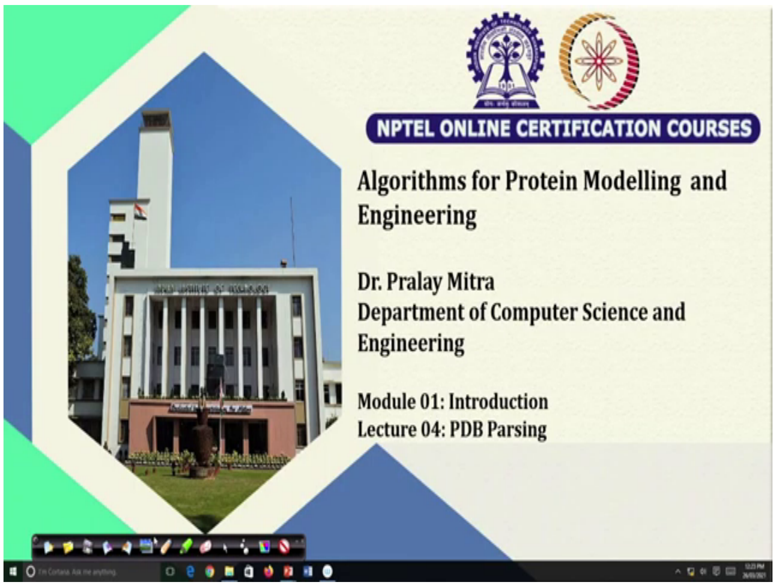
Lecture: 04
PDB Parsing

Welcome back.

We are going to discuss the PDB parsing. This is important because, in most of the algorithms that we will be developing and implementing next, one of the reliable sources for input data is PDB. Specifically, when we shall perform structure-based analysis or, we develop the algorithm that needs the protein structure, then we need to take the input from that protein data bank (PDB).

There are a few pros and cons - I should not say pros and cons rather, I will say that there are some limitations in the protein data structure. So, we should know the limitations and we should also know how to parse that data.

(Refer Slide Time: 01:13)

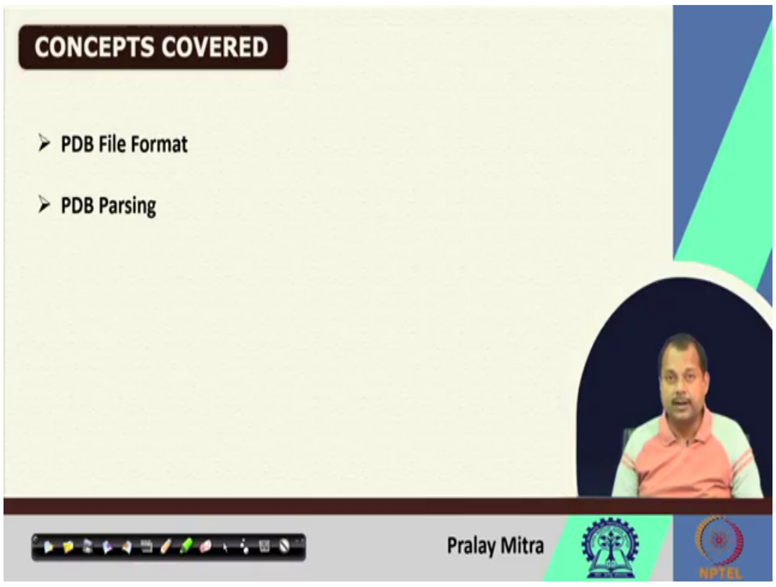


NPTEL ONLINE CERTIFICATION COURSES

Algorithms for Protein Modelling and Engineering

Dr. Pralay Mitra
Department of Computer Science and Engineering

Module 01: Introduction
Lecture 04: PDB Parsing



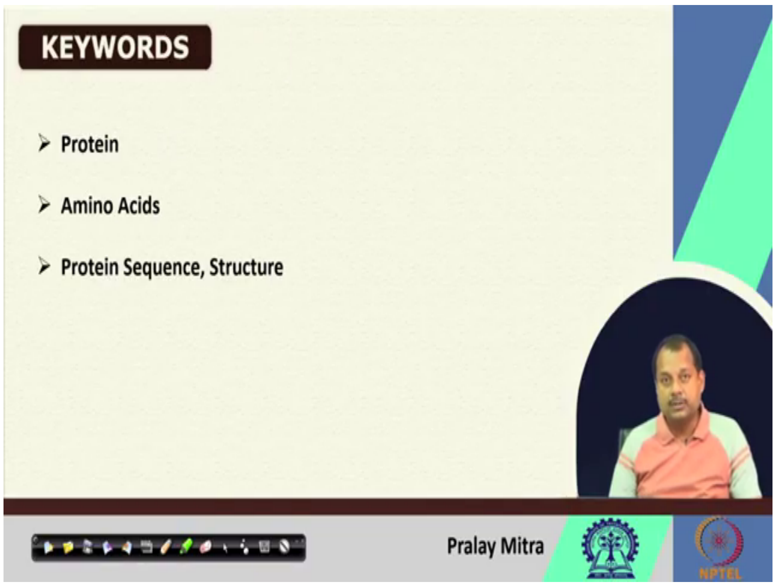
CONCEPTS COVERED

- PDB File Format
- PDB Parsing

Pralay Mitra

While parsing the data, we should focus on the fact that it is a flat-file. I mean only the column information is given and from that column information I have to extract the data. Currently, I shall not specify any specific programming languages, rather I shall explain it in pseudocode. But, you may find some similarities with the most widely used C programming language.

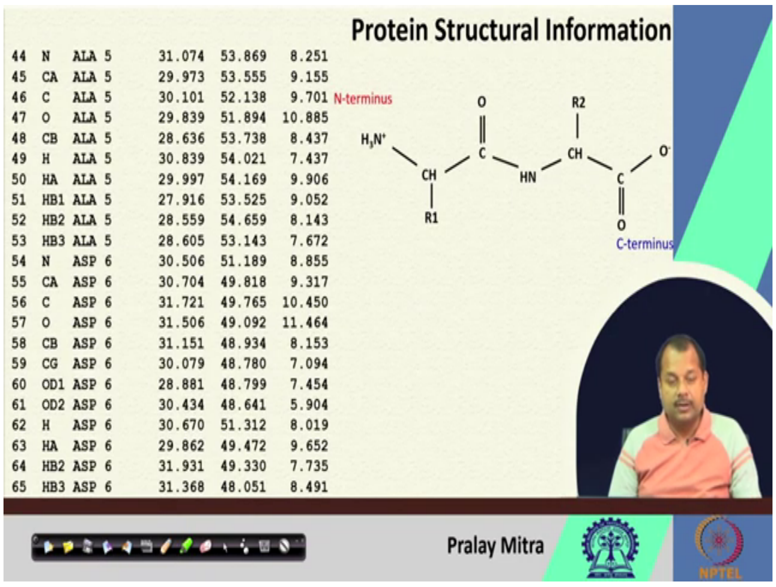
(Refer Slide Time: 01:57)



KEYWORDS

- Protein
- Amino Acids
- Protein Sequence, Structure

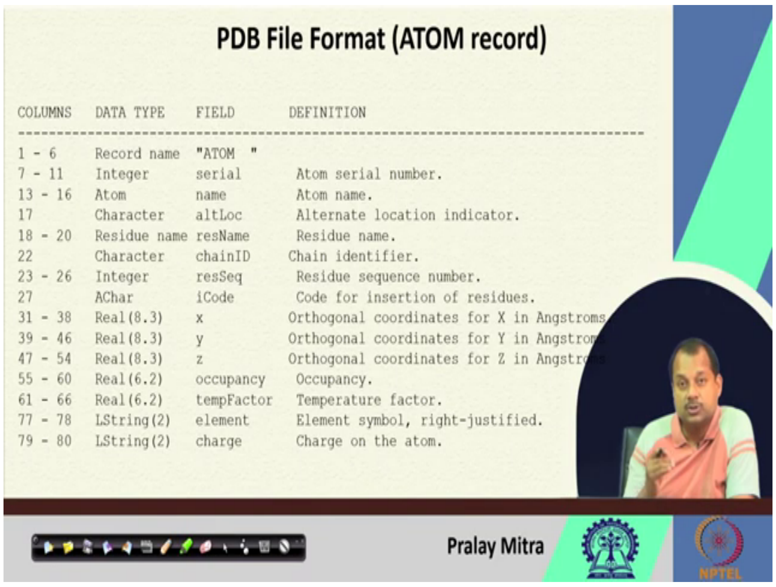
Pralay Mitra



Protein Structural Information

```
44  N    ALA 5     31.074  53.869   8.251
45  CA   ALA 5     29.973  53.555   9.155
46  C    ALA 5     30.101  52.138   9.701
47  O    ALA 5     29.839  51.894  10.885
48  CB   ALA 5     28.636  53.738   8.437
49  H    ALA 5     30.839  54.021   7.437
50  HA   ALA 5     29.997  54.169   9.906
51  HB1  ALA 5     27.916  53.525   9.052
52  HB2  ALA 5     28.559  54.659   8.143
53  HB3  ALA 5     28.605  53.143   7.672
54  N    ASP 6     30.506  51.189   8.855
55  CA   ASP 6     30.704  49.818   9.317
56  C    ASP 6     31.721  49.765  10.450
57  O    ASP 6     31.506  49.092  11.464
58  CB   ASP 6     31.151  48.934   8.153
59  CG   ASP 6     30.079  48.780   7.094
60  OD1  ASP 6     28.881  48.799   7.454
61  OD2  ASP 6     30.434  48.641   5.904
62  H    ASP 6     30.670  51.312   8.019
63  HA   ASP 6     29.862  49.472   9.652
64  HB2  ASP 6     31.931  49.330   7.735
65  HB3  ASP 6     31.368  48.051   8.491
```

Pralay Mitra



PDB File Format (ATOM record)

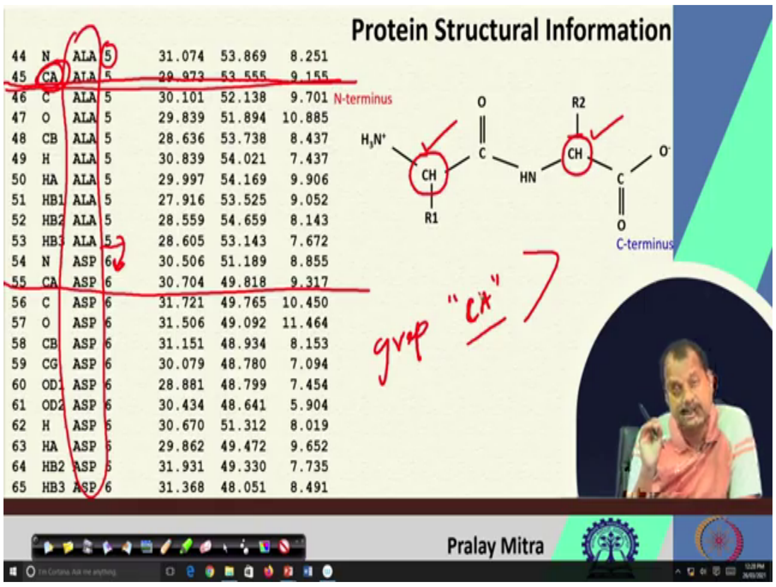
```
COLUMNS   DATA TYPE     FIELD        DEFINITION
------------------------------------------------------------------------------------
1 - 6     Record name   "ATOM  "
7 - 11    Integer       serial        Atom serial number.
13 - 16   Atom          name          Atom name.
17        Character     altLoc        Alternate location indicator.
18 - 20   Residue name  resName       Residue name.
22        Character     chainID      Chain identifier.
23 - 26   Integer       resSeq        Residue sequence number.
27        AChar         iCode         Code for insertion of residues.
31 - 38   Real(8.3)     x            Orthogonal coordinates for X in Angstroms
39 - 46   Real(8.3)     y            Orthogonal coordinates for Y in Angstron
47 - 54   Real(8.3)     z            Orthogonal coordinates for Z in Angstr
55 - 60   Real(6.2)     occupancy     Occupancy.
61 - 66   Real(6.2)     tempFactor    Temperature factor.
77 - 78   LString(2)    element       Element symbol, right-justified.
79 - 80   LString(2)    charge        Charge on the atom.
```

Pralay Mitra

And today we shall continue the PDB file format and shall discuss how-to scan the PDB file. Now, this is what we have discussed in the last class. And also, we discussed this particular column information, data type, field, and definition of that one.

The first column will have Atom information, after that is the serial number of the atom from columns 7 to 11. After that atom name, then it is alternate allocation, etcetera. Can you tell if I am interested to generate one sequence from this protein structure information then what I can do?

(Refer Slide Time: 02:48)



Protein Structural Information

```
44  N    ALA 5     31.074  53.869   8.251
45  CA   ALA 5     29.973  53.555   9.155
46  C    ALA 5     30.101  52.138   9.701
47  O    ALA 5     29.839  51.894  10.885
48  CB   ALA 5     28.636  53.738   8.437
49  H    ALA 5     30.839  54.021   7.437
50  HA   ALA 5     29.997  54.169   9.906
51  HB1  ALA 5     27.916  53.525   9.052
52  HB2  ALA 5     28.559  54.659   8.143
53  HB3  ALA 5     28.605  53.143   7.672
54  N    ASP 6     30.506  51.189   8.855
55  CA   ASP 6     30.704  49.818   9.317
56  C    ASP 6     31.721  49.765  10.450
57  O    ASP 6     31.506  49.092  11.464
58  CB   ASP 6     31.151  48.934   8.153
59  CG   ASP 6     30.079  48.780   7.094
60  OD1  ASP 6     28.881  48.799   7.454
61  OD2  ASP 6     30.434  48.641   5.904
62  H    ASP 6     30.670  51.312   8.019
63  HA   ASP 6     29.862  49.472   9.652
64  HB2  ASP 6     31.931  49.330   7.735
65  HB3  ASP 6     31.368  48.051   8.491
```

Structure information means this structure information. With the structure information I have, I need to look for this column. And from this column, I need to read that column information and from there I need to generate the FASTA sequence. How can I do that one? While doing that you should remember that this is representing the amino acid.

But in each row, one particular amino acid will many times be based upon how many atoms are there inside that amino acid. That way by row-scanning, it will be difficult to know how many amino acids are there? If I look at the residue/amino acid ID then I can identify it by noticing the change in ID.

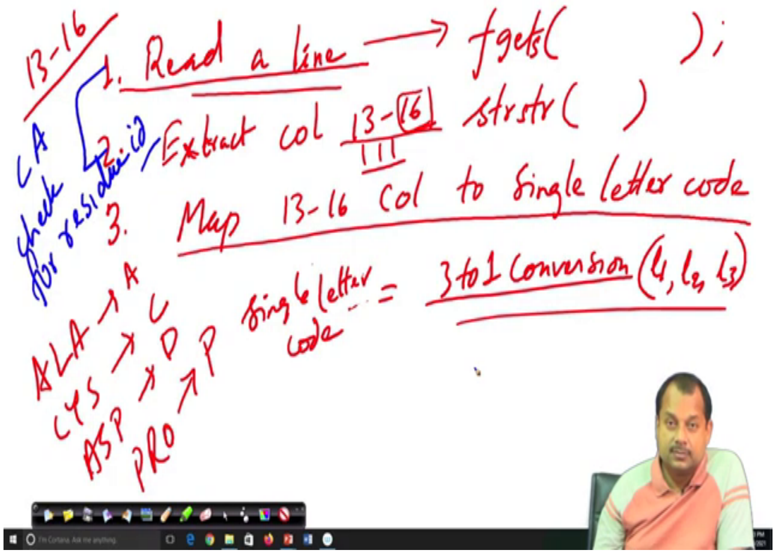
When I am scanning this file to know how many amino acids are there and who are they, then I need to identify the change in amino acids. Because after one alanine there can be another alanine and if there are multiple alanine molecules one after another, then whether it is the continuation of the alanine or the change of the alanine that I have to figure out.

One way to look at this is by noticing 5 changing to 6. Instead one can look at the row where the atom is CA. Because this CA, C alpha must be present in any amino acid unless there is not a huge structural information loss. Usually, if at least one atom is present corresponding to any amino acid then that is going to be C alpha (CA).

You should rely on this C alpha. And if you rely on this C alpha then the best thing is to look for this column while parsing. If you read only the C alpha and this C alpha then you will get the consecutive amino acids. If you are using a Linux or Unix environment then there is one command (at the terminal) called *grep*, which helps you to search for the existence of CA by outputting the lines consisting of CA.

Now, it may come to your mind if there is any other CA substring inside that line. Yes, that will create a problem. So, you have to be vigilant or you have to be careful about that fact. But this is one simple suggestion. This is my atom, this is my column corresponding to the atom. What I shall do with that column? This is columns 13 to 16. What shall I do? I shall read this 13 to 16. If I open this one then I mention 13 to 16. What I shall do is read one line. Next, extract 13 to 16, map 13 to 16 columns to single letter code. These are the steps. For each programming language, there will be one command for the read a line.

(Refer Slide Time: 06:23)



13-16

1. Read a line → fgets( );

2. Extract col 13-16 strstr( )

3. Map 13-16 col to single letter code

CA check for residue id

single letter code = 3 to 1 conversion (l_1, l_2, l_3)

ALA → A

CYS → C

ASP → D

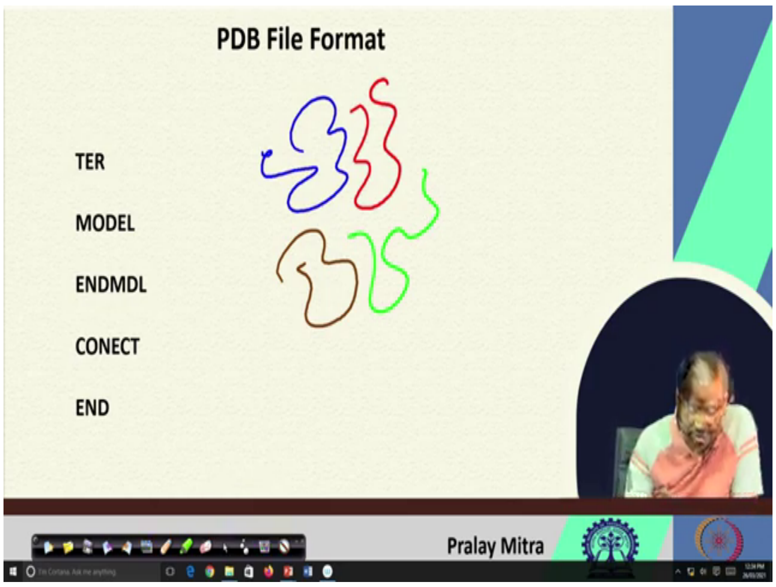
PRO → P

In the C programming language, you have *fgets*() for reading from the line. After reading that you can extract using say *strstr*() string function. Else you can read columns 13, 14, 15, and 16 - basically 13, 14, and 15. Here 16 will be extra that we shall discuss later when the multiple atoms/residues situation will arise. The situation, when experimentally or computationally, it is not possible to confirm whether it is going to be residue 1 or residue 2 (like leucine or isoleucine). Then instead of keeping one reside information, both are kept (like leucine and isoleucine) and mention that there is a 50% probability that it can be isoleucine and 50% that it will be leucine.

Column 16 is used for that. Ignoring that you will get a three-letter code. Therefore, you require one function (viz., *three2one*()) to covert a three-letter amino acid code to a one-letter amino acid code. This *three2one*() function will accept three-letter code from columns l8 to 20 and will return a single letter code. Thus, ALA is going to be A, CYS is going to be C, ASP is going to be D, PRO is going to be P, and so on. To summarize, you read a line, extract columns 18 to 20, and map to a single letter code.

In between, two read a line, extract 18 to 20. You have to come up with something so that either you read CA or you check for residue ID. If you do not check for the residue ID or CA then you will get multiple instances of one particular amino acid, which is not correct. This way what you are getting is one particular protein sequence for the structure you parse.

(Refer Slide Time: 10:57)



PDB File Format

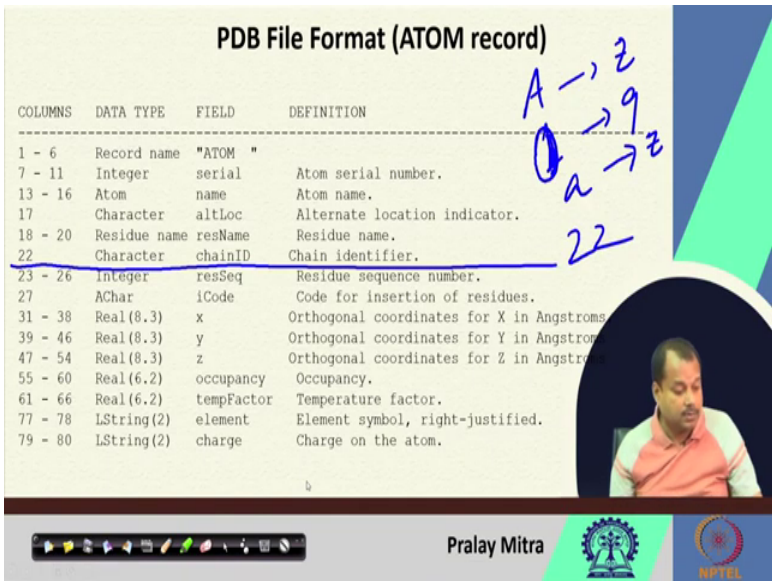
TER

MODEL

ENDMDL

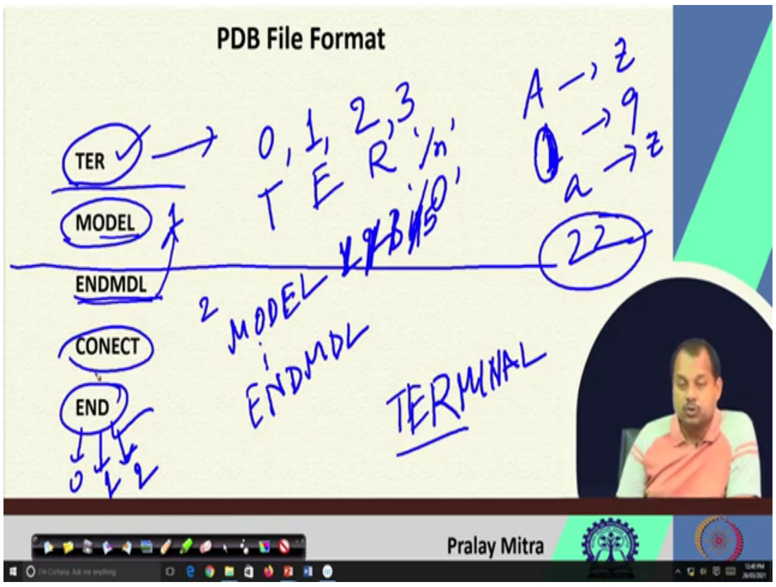
CONECT

END



PDB File Format (ATOM record)

```
COLUMNS   DATA TYPE     FIELD        DEFINITION
-------------------------------------------------------------------------------------
 1 - 6    Record name   "ATOM  "
 7 - 11   Integer       serial        Atom serial number.
13 - 16   Atom          name          Atom name.
17        Character     altLoc        Alternate location indicator.
18 - 20   Residue name  resName       Residue name.
22        Character     chainID      Chain identifier.
23 - 26   Integer       resSeq        Residue sequence number.
27        AChar         iCode         Code for insertion of residues.
31 - 38   Real(8.3)     x            Orthogonal coordinates for X in Angstroms.
39 - 46   Real(8.3)     y            Orthogonal coordinates for Y in Angstroms.
47 - 54   Real(8.3)     z            Orthogonal coordinates for Z in Angstroms.
55 - 60   Real(6.2)     occupancy     Occupancy.
61 - 66   Real(6.2)     tempFactor    Temperature factor.
77 - 78   LString(2)    element       Element symbol, right-justified.
79 - 80   LString(2)    charge        Charge on the atom.
```



PDB File Format

TER

MODEL

ENDMDL

CONECT

END

After getting this protein sequence what else is interesting to you, I shall come to that. I am interested to know how many chains are there or how many connected components are there. When I say how many connected components are there, you remember that I have a drawing, not the same but something like these two chains, I may have three, I may have four.

Each one is one connected component. I mentioned that one chain ID will be given or mentioned that chain ID you can have here and that chain ID you will have this here 22, you are getting gene ID at 22. What you need to do is to read this character from column 22.

Again, using any programming language, you can read that column 22, which is going to be your chain ID and I mentioned that chain ID can be either a character from A to Z, or a single digit 0 to 9 (considered as a character). It may be a small letter or a capital letter. Sometimes you will find that there are two chains both are with the same ID like A and A. If so then there must be TER keyword indicating the end of a chain. TER starts at the beginning of the line.

Therefore, using any programming language you can read your zeroth column, first column, second column, and optionally say the third column to check whether it is TER (short form of TERMINAL). If it is only TER, then this is T, this is E, this is R, and this is either newline ('\n') or end of word '\0' (in C programming language). In both cases, it indicates the end of one chain - the end of one connected component.

Therefore, by counting these numbers of TER, you may get how many chains are there or what is the end of the chain. Also, I am suggesting keeping an eye on column number 22 and use both. Because I have seen instances where there is no TER. I have seen instances where there are multiple chains with the same ID. So, to avoid those, it will be better to use both the information.

Regarding this END keyword, it indicates the end of the coordinate information. Once you will encounter this END, you know the end of the file. Again this is 0, this is 1, this is 2 column number which means, it will start from the new line.

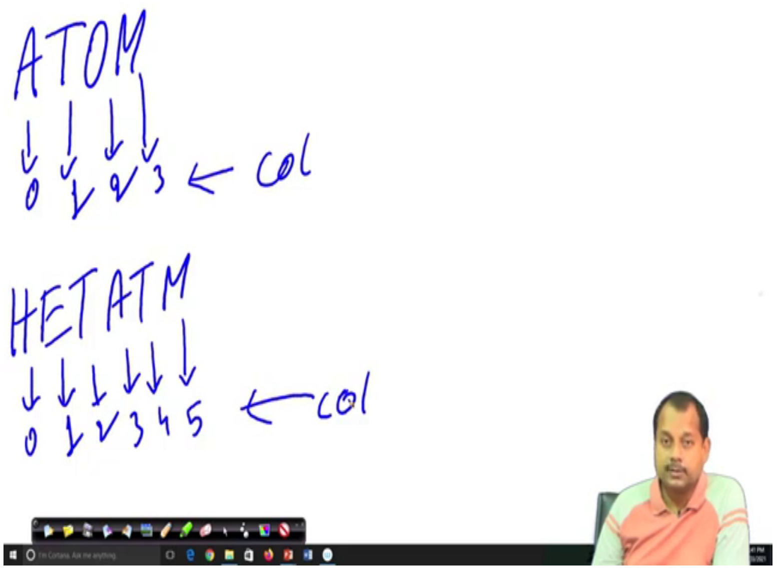
MODEL information, as I mentioned occurs mostly in the NMR structure. If several instances of one structure are there and the author wishes to keep all such instances, then different instances will be separated by their MODEL information. MODEL starts from the new line. After MODEL there will be one integer value say 1, 2, 3 that indicates how many models are there. This integer value on the model number - 1, 2, etcetera up to say 18, 20 whatever is there it will be.

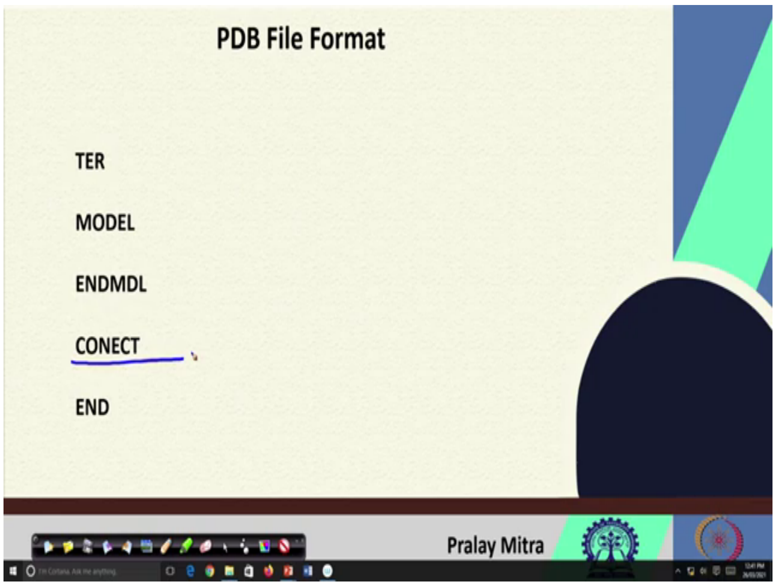
ENDMDL indicates the end of the MODEL which again starts from the first column of a new line. End of which model? The model which has started. Thus, corresponding to each model there will be pair information in MODEL indicating only one MODEL. If I have five MODELs, they need to be starting with a MODEL 1, then there will be 1 ENDMDL, after that one there will be MODEL 2, ENDMDL, after that 3, after that 4, after that 5 and ENDMDL will be there.

Regarding CONECT keyword, I mentioned in my last lecture that for the essential amino acids, I know what are the covalent bonds. And since that list is limited, so all my information I can keep in my program and that is why the covalent information for the essential amino acids are not explicitly present in the PDB structure. However, if there are some nonstandard amino acids or say other molecules starting from the water molecule or any moiety or small molecule, etcetera. It is not possible to remember what are their covalent bonds. The best idea is to declare them as heteroatom to separate them from the Atom information. Who is the atom? I am assuming that is part of the essential amino acids for which I know the covalent bond information.

And when it is the heteroatom for which implicit information is not there. Therefore, explicitly I have to mention that what is their CONECT information, means that what are their covenant bonds and for that, I will be using their Atomic serial number. Of course, they will be represented as heteroatom instead of atoms.

(Refer Slide Time: 18:51)





PDB File Format

TER

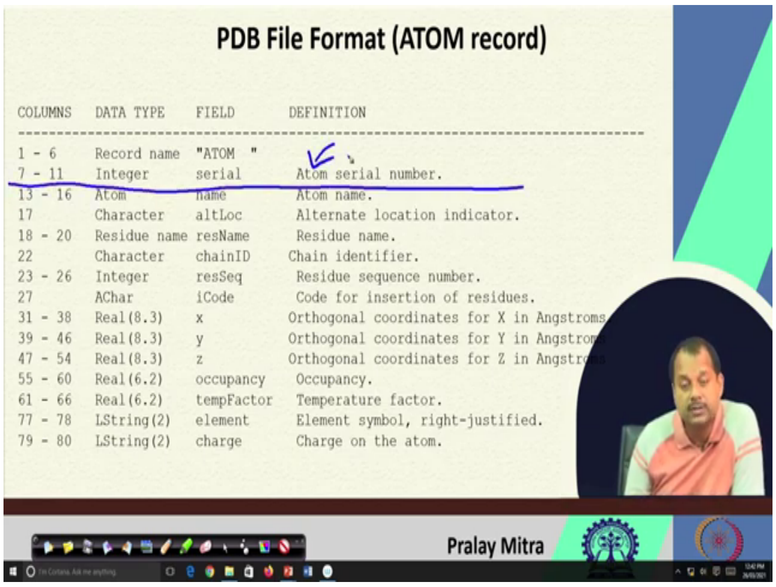
MODEL

ENDMDL

CONECT

END

Pralay Mitra



PDB File Format (ATOM record)

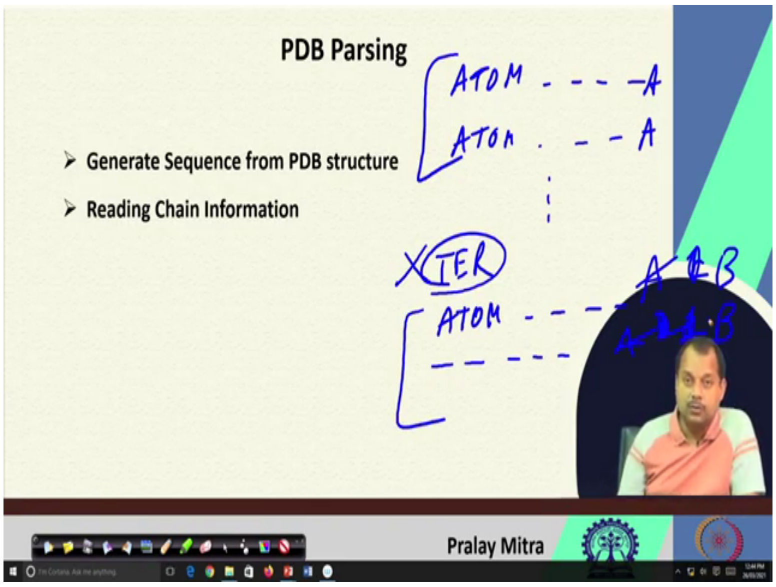
COLUMNS	DATA TYPE	FIELD	DEFINITION
1 - 6	Record name	"ATOM "	
7 - 11	Integer	serial	Atom serial number.
13 - 16	Atom	name	Atom name.
17	Character	altLoc	Alternate location indicator.
18 - 20	Residue name	resName	Residue name.
22	Character	chainID	Chain identifier.
23 - 26	Integer	resSeq	Residue sequence number.
27	AChar	iCode	Code for insertion of residues.
31 - 38	Real(8.3)	x	Orthogonal coordinates for X in Angstroms
39 - 46	Real(8.3)	y	Orthogonal coordinates for Y in Angstroms
47 - 54	Real(8.3)	z	Orthogonal coordinates for Z in Angstroms
55 - 60	Real(6.2)	occupancy	Occupancy.
61 - 66	Real(6.2)	tempFactor	Temperature factor.
77 - 78	LString(2)	element	Element symbol, right-justified.
79 - 80	LString(2)	charge	Charge on the atom.

Pralay Mitra

If it is an essential (one of twenty that I mentioned) amino acid then each line will start with an ATOM. This is my 0 position, this is 1, this is 2, this is 3, and this is the column information. In the case of other (small molecules or non-standard amino acids), then it is HETATM that starts at 0, this is 1, this is 2, this is 3, this is 4, this is 5, and this is my column information. And, for this one, I need the CONECT information.

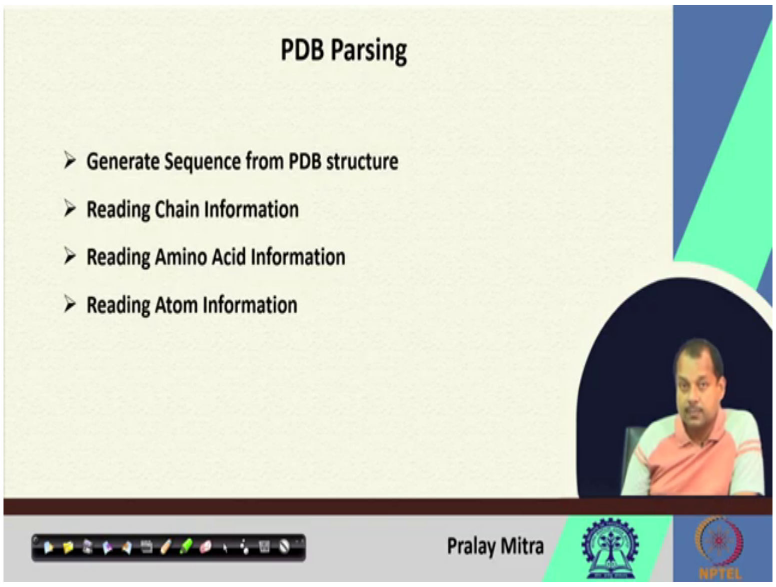
The CONECT information indicates who will be connected with what. But definitely rest of the thing say, for example, this integer, the serial number. Although it is mentioned that the atom serial number, for the hetero atom also one serial number will be there and that serial number information will be used to know the CONECT information.

(Refer Slide Time: 20:20)



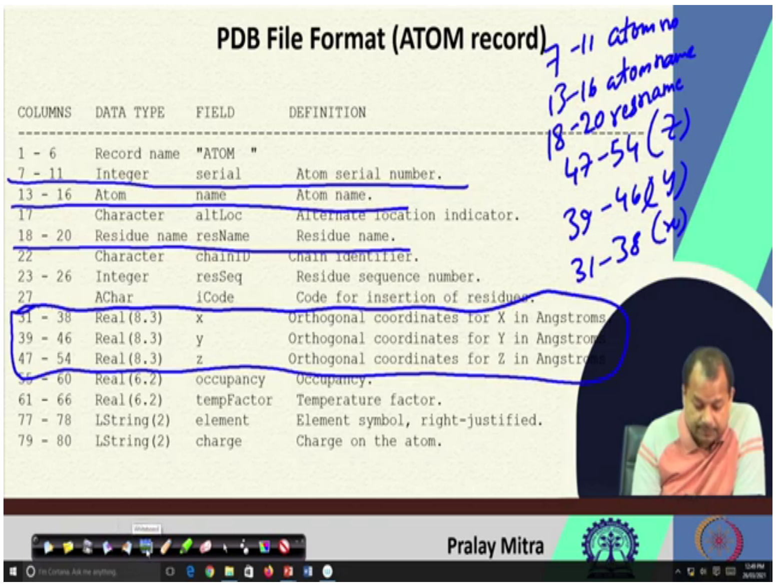
PDB Parsing

- Generate Sequence from PDB structure
- Reading Chain Information



PDB Parsing

- Generate Sequence from PDB structure
- Reading Chain Information
- Reading Amino Acid Information
- Reading Atom Information



PDB File Format (ATOM record)

COLUMNS	DATA TYPE	FIELD	DEFINITION
1 - 6	Record name	"ATOM "	
7 - 11	Integer	serial	Atom serial number.
13 - 16	Atom	name	Atom name.
17	Character	altLoc	Alternate location indicator.
18 - 20	Residue name	resName	Residue name.
22	Character	chainID	Chain identifier.
23 - 26	Integer	resSeq	Residue sequence number.
27	AChar	iCode	Code for insertion of residues.
31 - 38	Real(8.3)	x	Orthogonal coordinates for X in Angstroms
39 - 46	Real(8.3)	y	Orthogonal coordinates for Y in Angstroms
47 - 54	Real(8.3)	z	Orthogonal coordinates for Z in Angstroms
55 - 60	Real(6.2)	occupancy	Occupancy.
61 - 66	Real(6.2)	tempFactor	Temperature factor.
77 - 78	LString(2)	element	Element symbol, right-justified.
79 - 80	LString(2)	charge	Charge on the atom.

7-11 atom no
13-16 atom name
18-20 resname
47-54 (z)
39-46 (y)
31-38 (x)

Pralay Mitra

We discussed how to generate a sequence from the PDB structure. And on the last lecture, I also mentioned that the sequence you generated from the PDB structure I mean the atomic coordinate information may not match with the SEQRES information it has in it. Because some residue structure information of some residue may be missing and if it is then you may not track it down.

Next, we discussed reading the chain information, where you may go by the chain ID which is a piece of single character information either A to Z or a to z or 0 to 9 or you may go by the TER based information. You may probably have both in your program so that you can take whatever is available. In some cases you may see TER is missing, in some cases, you may see that TER is there but multiple chains are with the same ID, so it is possible.

If the situation is something like I have Atom information ATOM and say chain ID is also mentioned here A and this and after that I have TER, then ATOM, then A, again A, if like that I go then using this TER you can identify one chain, and this is another chain.

But if TER information is missing then usually this A will not continue, then this A will be 1. But I am not telling that if the TER information is there, then always it will be AAA it may be AAA, sorry it will be 1, so 1 1 or say maybe BBB or may be different. That is why I am suggesting you have both TER as well as the change in column number 22, that is containing the information of your chain.

Next is reading the amino acid information. When I am reading the protein sequence or say I am generating the protein sequence from the PDB, then I am also reading the amino acid information. That is the information corresponding to which amino acid is present. But apart from that, if I need a little more information like I know that since it is only 20 amino acids.

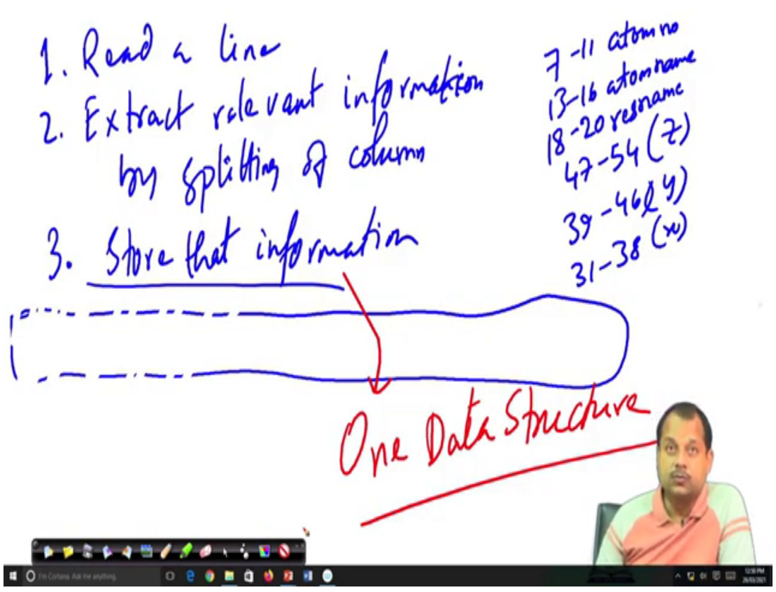
It is very much possible to have one lookup table or one separate data structure or file where I shall have the full connectivity and atomic information of each amino acid, not the coordinate I am talking about. I am talking about how many atoms are there corresponding to each amino acid and which are they what are their covalent bonds and if along with that one I wish to attach some physical-chemical information, then those things I can have.

Let us assume that I wish to make a match with the aspartic acid, whether the O^- is present in the side chain or that particular atom is missing. If I need that kind of information, then I have to write one separate function for reading amino acid information. Next is reading the ATOM information. Combining the amino acid and ATOM information is very essential. Because when I am reading the ATOM information, then I am reading the coordinate information. Coordinate information is available at the atomic level.

When I say it is the residue, then unless I mentioned at the atomic level that the C alpha atom of the residue is representing that residue or the C beta atom, I cannot say that I can able to read the atomic information for the residue level. The coordinate information is available only at the atomic level.

Residue level coordinate information does not make sense unless otherwise, I mention that which atom is there. I am reading the Atom information. Let me go back and give you this information first. The coordinate information is available here. This means 31 to 38 is X coordinate then 39 to 46 is Y coordinate and 47 to 54 is Z coordinate. Along with I need 7 to 11 atom number, then 13 to 16 atom name and I also need this 18 to 20 residue name.

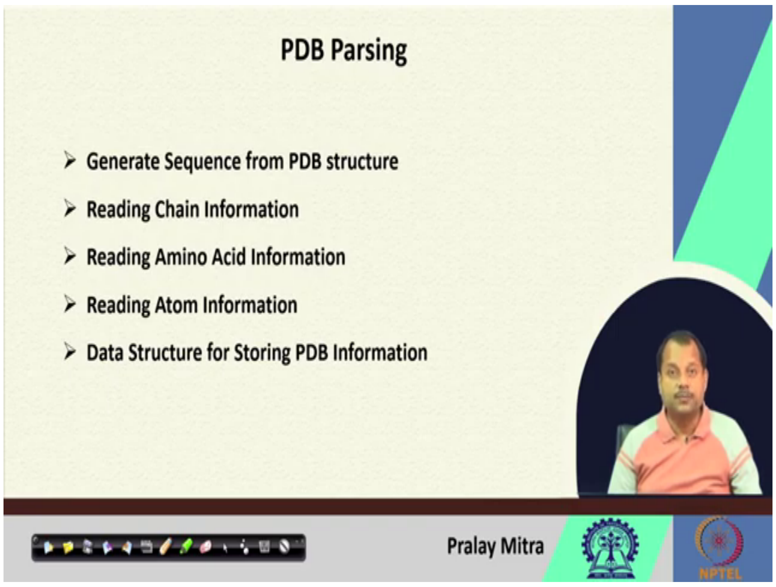
(Refer Slide Time: 26:55)



1. Read a line
2. Extract relevant information by splitting of column
3. Store that information

7-11 atom no
13-16 atom name
18-20 resname
47-54 (z)
39-46 (y)
31-38 (x)

One Data Structure



PDB Parsing

- Generate Sequence from PDB structure
- Reading Chain Information
- Reading Amino Acid Information
- Reading Atom Information
- Data Structure for Storing PDB Information

Pralay Mitra

REFERENCES

- https://www.rcsb.org/
- https://www.wwpdb.org/documentation/file-format

Pralay Mitra

What can I do using this information?. Read a line using some function, extract relevant information by of column, and store that information. Of course, I need some data structure to store that information. Defining the data structure is going to be very important. The data structure you should define in such a way that your data access will be fast enough.

These we shall discuss from week two onwards. From week two onwards we need to store the data and process the data. We should remember that while reading the ATOM information and the amino acid information, we need to define one data structure also for storing the PDB information. As of now, I discussed some of the topics of PDB parsing. Now, you can practice writing the code for PDB parsing since PDB information will be required for us when we will be developing the algorithms. Mostly the input will be taken from the PDB and also it will be taken from the UniProt. Please note that UniProt provides the sequence data. Thank you.