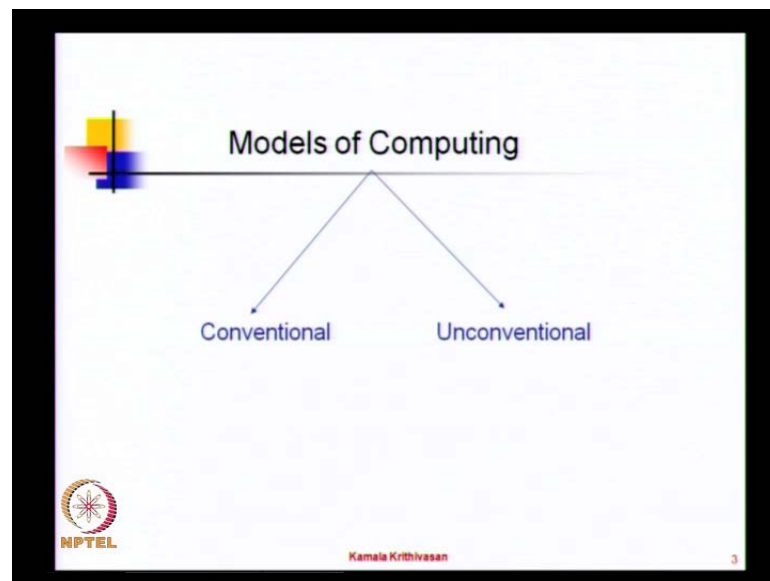


Theory of Computation
Prof. Kamala Krithivasan
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture No. # 42
Membrane Computing

Today, we shall consider a new paradigm of computing namely membrane computing.

(Refer Slide Time: 00:19)



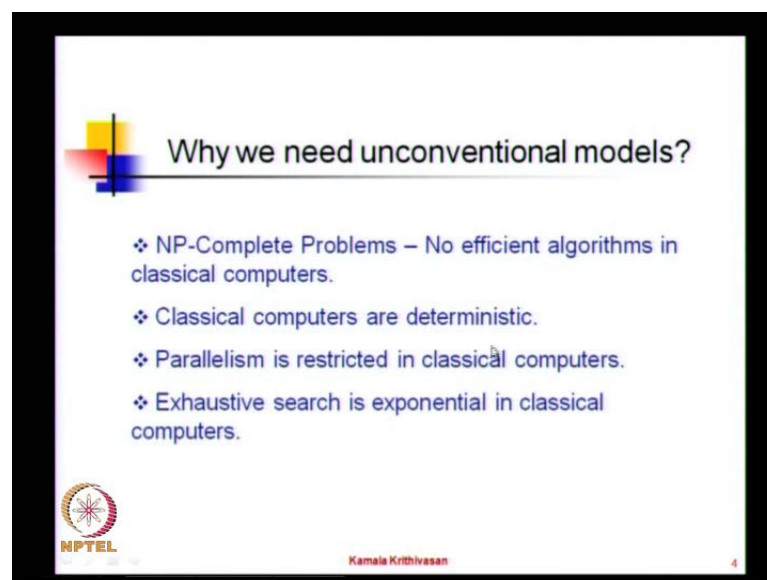
So, the models of computing can be considered as conventional models and unconventional models. What are the **conventional** models of computing? The Turing machine is a conventional model of computing; in 1936, Turing defined a Turing machine, and he was able to foresee until what is possible with the computer and what is not possible with a computer even before 15 years the computer was built. And whatever he has said it is called Church-Turing hypothesis and it has stood the test of time, but

Turing machine we have already seen what it is and all that it is not random access.

So, simulate the current basis system in an exact manner register machines have been defined; so they are also conventional models of computing, the connection between register machines and Turing machines have been studied time wise. If something can be done in one T units of time, at the most it may take T squared or T cube units of time in the other. So, polynomial or non-polynomial time will not be affected by simulating one model with the other, one model is the Turing machine, the other model is the random access machine.

Now in contrast to this we have the unconventional models of computing, what are the unconventional models of computing? Paradigms like DNA computing, membrane computing, quantum computing, they are unconventional models of computing. So, this have come into existence in the recent times and we have found that certain problems can be tackled in a very efficient manner using this, but the practical use of this is still a question and is being explored.

(Refer Slide Time: 02:37)



The slide features a title 'Why we need unconventional models?' with a decorative graphic of overlapping colored squares (yellow, red, blue) to the left. Below the title is a list of four bullet points, each starting with a blue diamond symbol. At the bottom left is the NPTEL logo, and at the bottom center is the name 'Kamala Krithivasan'.

Why we need unconventional models?

- ❖ NP-Complete Problems – No efficient algorithms in classical computers.
- ❖ Classical computers are deterministic.
- ❖ Parallelism is restricted in classical computers.
- ❖ Exhaustive search is exponential in classical computers.

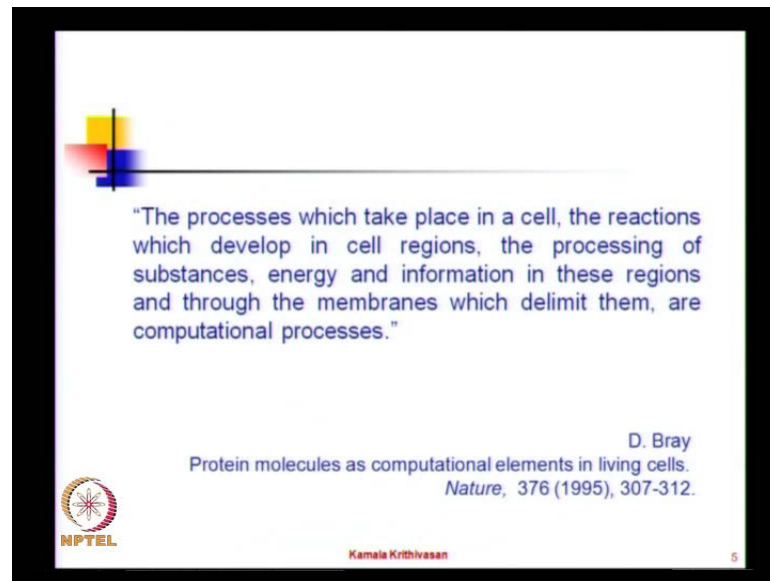
NPTEL
Kamala Krithivasan

Now, why we have to study unconventional models? What is the need when a already existing models like Turing machines etcetera are good enough. We have already seen,

what is meant by a NP-complete problem? They do not seem to have efficient deterministic polynomial time algorithm, so no efficient algorithms in classical computers and the classical computers are deterministic in nature, they can do step by step some operation, even though today, we talk about parallel computers we have the number of processes bounded, so in a parallel machine you may have 64 processes, whatever it is it is bounded by a number n .

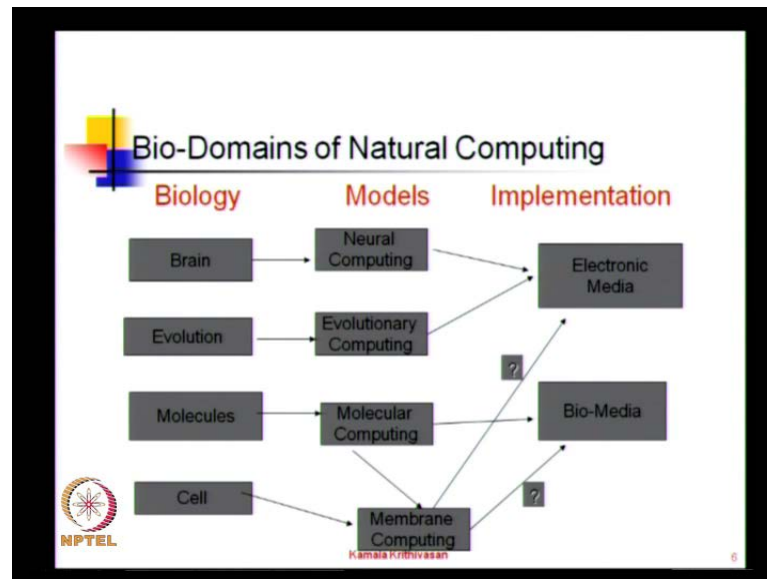
So, the problem may be divided into 16 sub problems and given to the sub processes and they can work simultaneously on the problem, but when the problem size increases the number of processes cannot be increased you have 16 processes, it means you always have 16, you cannot make it 64, then 128 and so on, you cannot increase the number of processes right. Whereas, in the new models of computation like DNA computing and membrane **computing**, unbounded parallelism is there because of the unbounded parallelism, you can have DNA strands and they can exist within 1 cubic centimeter of a solution, millions of DNA strands can exist. And because of that the action takes place parallel and you can have a number of operations performed simultaneously, you have a unbounded parallelism there, that is why we tried to have unconventional models, so that some of the NP complete problems can be solved in linear time or in polynomial time. Whenever we have to make exhaustive search then it is always exponential in the classical computers, but by making use of the unbounded parallelism and in DNA computing we also have the Watson-crick complementarity, because of those things we can try to reduce the time from exponential to polynomial.

(Refer Slide Time: 05:21)



Today, we shall see about membrane computing, it is a bio-inspired model or it is a model of natural computing, but it is inspired by biology. So, what is inspiration? The process, which takes place in a cell the reactions which develop in cell regions, the processing of substances energy and information in these regions and through the membranes, which delimit them are the computational processes. And we try to simulate these processes and see that anything we can define in a formal manner using this model of computation.

(Refer Slide Time: 06:07)



So, if you look into the different branches of natural computing, you can divide them like this models inspired by nature, that is bio-inspired computing, under this comes Neural computing you know that nowadays Neural networks play a very important role in the field of computer science and they have come into existence because of the simulation of brain. In Neural computing you try to simulate the way the brain works and the Neural networks are implemented with the electronic silicon computer. So, the implementation is through the Electronic media. And you also have what is known as an Evolutionary computing genetic algorithms, they try to solve an algorithm making use of representing an instance using a bit string and then they perform the operations of cross over mutation, selection, etcetera, which happens in the actual evolution of humans or any other organism in nature.

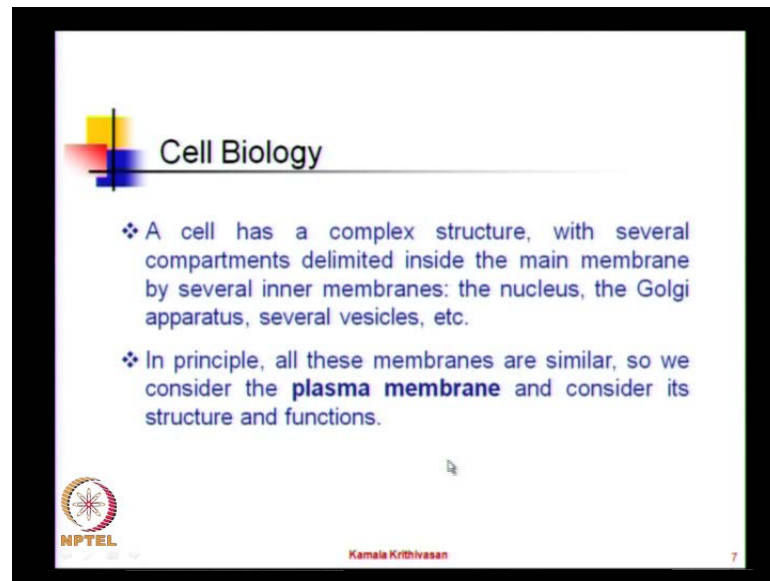
And because they simulate the evolution it is called Evolutionary computing and genetic algorithms come under this, genetic algorithms are mostly heuristic algorithms and even though they are heuristic algorithms in most cases they give very good solutions, to some problems because they converge, even though the convergence cannot be properly proved. In most of the examples where it is considered it is found that by taking the model in a suitable manner and representing the strings in a suitable manner the conversions can be achieved, even for network algorithms like routing etcetera,

Evolutionary computing has been used and it is a very good area of research nowadays.

You also have what is known as swarm intelligence ((C)) and Ant colony paradigms and so on, which are biologically inspired. In contrast to that you also have some models which simulate nature and show how nature is being developed like fractal geometry and cellular automata, you might have heard about the game of life and so on, how it is done with cellular automata is second one. The third one is actually computing with Molecules or computing with natural objects that comes under Molecular computing, the actual thing is the molecules and how they behave and interact with each other given solutions for problems and that is called Molecular computing, DNA computing, Peptide computing, they all come under this and the implementation is really using the DNA strands or the Peptides and the Molecules.




So, the implementation is through bio-media. In 1998 the paradigm membrane computing was proposed because it tried to simulate the behavior of what happens in a cell, the proteins, the enzymes, the membrane structure, how they develop, they pass through the membranes changing from one form to another and. So on. And initially the theoretical model was developed and the first paper in this area was published in 2000, at that time it was thought that there are two possibilities. One it could be implemented using the silicon computer, like evolutionary algorithms and so on. The other one is you may actually use membranes or cells and try to simulate the algorithms, this was the question in 2000 and in 2010 now this looks that it is not possible at all, how to make the algorithms be simulated by actual cells. Whereas, this looks more promising that is, this may develop as a bio-inspired model of computing, like Evolutionary computing or Neural computing and will be implemented in the electronic media only.

(Refer Slide Time: 10:53)



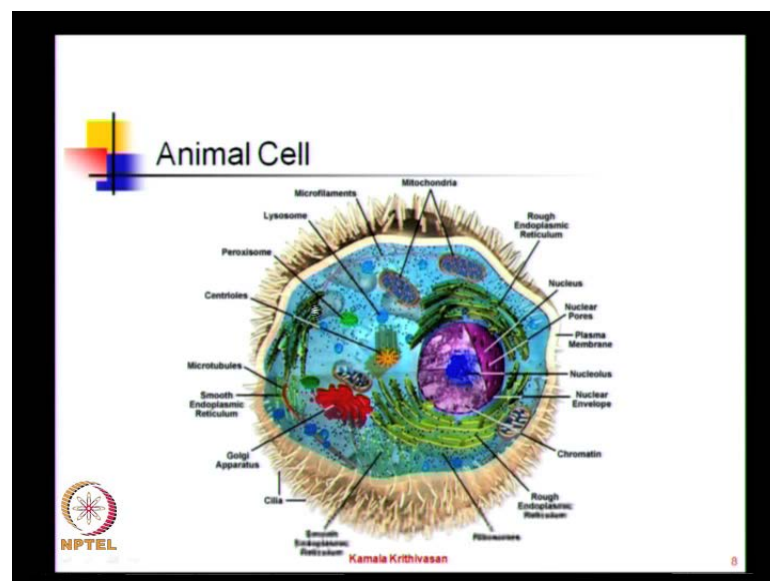
Cell Biology

- ❖ A cell has a complex structure, with several compartments delimited inside the main membrane by several inner membranes: the nucleus, the Golgi apparatus, several vesicles, etc.
- ❖ In principle, all these membranes are similar, so we consider the **plasma membrane** and consider its structure and functions.

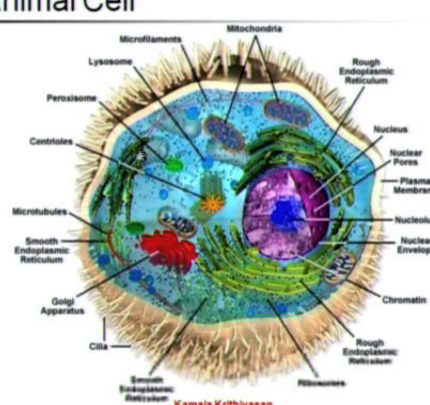
  




So, how do we go about defining and what is the motivation behind this new paradigm of computing? A cell has a complex structure with several compartments delimited inside the main membrane by several inner membranes, the nucleus, the Golgi apparatus several vesicles and so, on. In principle these membranes are all similar; so we can consider the plasma membrane and its functions.

(Refer Slide Time: 11:27)



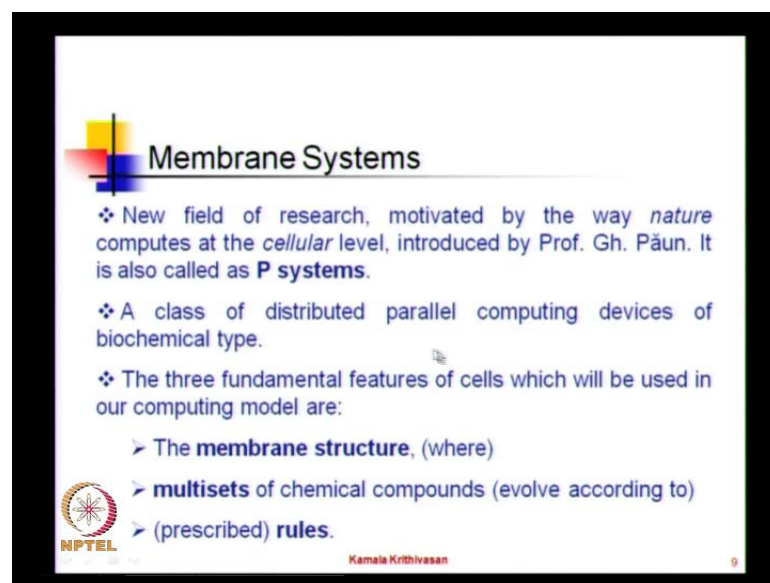
Animal Cell




A membrane in an animal cell looks like this, you can look at it as a big membrane inside which it has got several other things, this may be looked at as another membrane inside it something happens, this may be looked at as another membrane, certain things happen here like chromosomes or protein molecules develop and they pass through the membrane to the outer region or from the outer region they come into the inner region and some reactions take place, this is modeled in membrane computing.

(Refer Slide Time: 12:01)



Membrane Systems

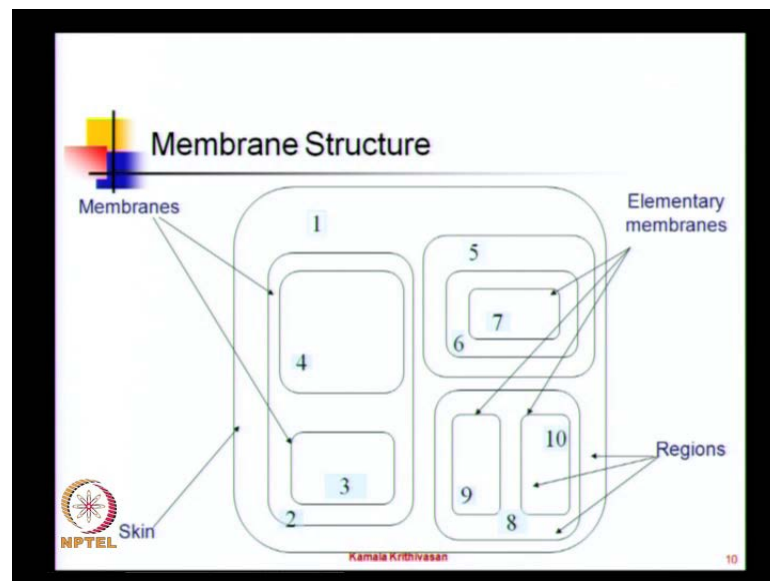
- ❖ New field of research, motivated by the way *nature* computes at the *cellular* level, introduced by Prof. Gh. Păun. It is also called as **P systems**.
- ❖ A class of distributed parallel computing devices of biochemical type.
- ❖ The three fundamental features of cells which will be used in our computing model are:
 - The **membrane structure**, (where)
 - **multisets** of chemical compounds (evolve according to)
 - (prescribed) **rules**.

 NPTEL Kamala Krithivasan 9

So, the membrane systems is a new field of research started in 98 and first paper published in 2000, it is motivated by the way nature computes at the cellular level and this was introduced by professor George Paun of Romania. And because of that it is called as P systems, this is the class of distributed parallel computing devices, because whatever happens within the membranes occurs is parallel and then messages pass from inside to outside of each membrane and while crossing the membranes they carry some information, so you can look at them as distributed model as well, so it is a distributed parallel model of computing. There are three fundamental features which have to be taken into account and they are the membrane structure that is a structure of the membrane. And the second one is the multisets that is these are the proteins or the enzymes or the chromosomes inside the membrane, you can look at them as chemical compounds and they are represented as multisets, later on we will see that you can look


at them as strings, arrays, etcetera. And these multisets of objects evolve, they change from time to time passing through the membranes and so on and that is controlled by a prescribed set of rules. So, in the paradigm has three components the structure, the multisite of objects and the rules.

(Refer Slide Time: 13:39)




A membrane has the following structure, the outer most membrane is called the Skin membrane within that you have membranes, this is membrane 2 is inside membrane 1, 1 is the outer most membrane or the Skin membrane, **membrane** 1, 5, 8 are inside this and within membrane 1 you have membrane 2, within membrane 2 you have 3 and 4, and within 5 you have 6, and within 6 you have 7, within 8 you have 9 and 10, these are the membranes, the outer most one is called the Skin. And a membrane which does not have any other membrane inside it is called an Elementary membrane and the region enclosed by a membrane, are considered. And some of the membrane will be considered as output membranes we will see what it is.

(Refer Slide Time: 14:42)



Objects

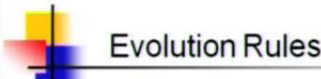
- ❖ Objects can be considered as *atomic* or they can have *structure*.
- ❖ Symbol-objects
- ❖ String-objects
- ❖ Array-objects
- ❖ Graph-objects



Kamala Krithivasan 11


The objects inside the membrane are atomic in nature and they have some structure, you can consider them as Symbol objects, this is the one which we will consider mainly, you can also have String objects. Recently Array objects and Graph objects have also been considered.

(Refer Slide Time: 15:03)




Evolution Rules

- ❖ Generally, we consider **context-free** [1] rules for processing both symbol-objects and string-objects.
- ❖ For string-objects, we may consider **splicing** [2] rules also.
- ❖ P systems with symbol-objects.
- ❖ Rewriting P systems.
- ❖ Splicing P systems.



[1] J. E. Hopcroft, J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley, 1979.
[2] Gh. Păun, G. Rozenberg, A. Salomaa, *DNA Computing. New Computing Paradigms*, Springer-Verlag, Berlin, 1998.

Kamala Krithivasan 12



Evolution Rules

- ❖ Generally, we consider **context-free** [1] rules for processing both symbol-objects and string-objects.
- ❖ For string-objects, we may consider **splicing** [2] rules also.
- ❖ P systems with symbol-objects.
- ❖ Rewriting P systems.
- ❖ Splicing P systems.


[1] J.E. Hopcroft, J.D. Ullman, *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley, 1979.
 [2] Gh. Păun, G. Rozenberg, A. Salomaa, *DNA Computing, New Computing Paradigms*, Springer-Verlag, Berlin, 1998.

Kamala Krithivasan

12

Generally context free rules are used in many cases context sensitive rules are also used for processing string objects. And symbol objects you have catalyst and sometimes cooperating rules are used, sometimes for strings you also use splicing rules and the symbol objects usually two or three objects can combine and form one object, so that is also allowed. For strings you use what is known as rewriting P systems, you can also use what is known as splicing P systems.

(Refer Slide Time: 15:49)



Formal Definition

P system of degree n , $n \geq 1$, is a construct $\Pi = (V, T, C, \mu, w_1, \dots, w_n, (R_1, \rho_1), \dots, (R_n, \rho_n), i_0)$.

where:

- > V is an alphabet; its elements are called *objects*;
- > $T \subseteq V$ is an output alphabet;
- > $C \subseteq V$, $C \cap T = \emptyset$, is a set of catalysts;
- > μ is a membrane structure;
- > w_i , $1 \leq i \leq n$, is a multiset of objects over V present in region i ;
- > R_i , $1 \leq i \leq n$, is a finite set of rules associated with region i ;
- > ρ_i , $1 \leq i \leq n$, is a partial order relation over R_i ;
- > i_0 is an output membrane.

Kamala Krithivasan

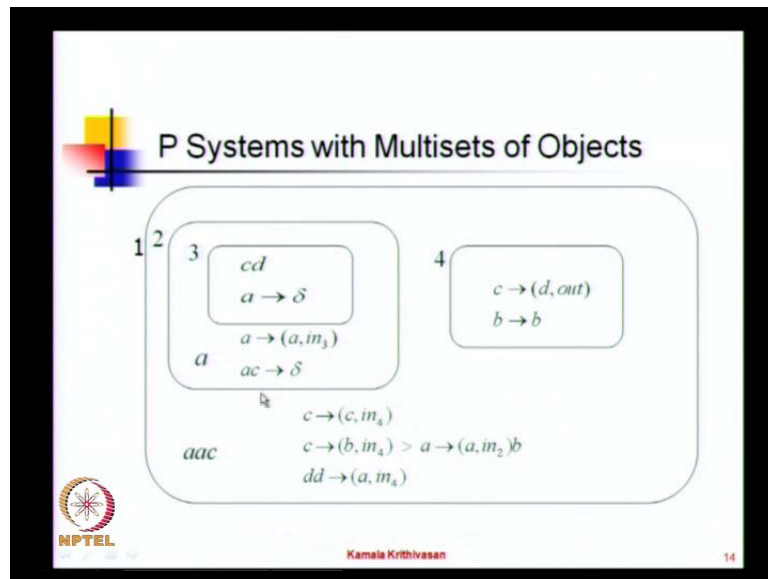
13

We shall see the basic model and the definition. The formal definition of a P system or a membrane is like this, it is of degree n means there are n membranes, where you have V is the total alphabet it is a objects, T is a subset of V which is the output alphabet, C is the set of catalyst. The initial definitions contained the use of catalyst and **later** definitions they do not consider catalyst separately, rather the use some sort of context sensitive rules. Actually catalyst encourage the development of some objects, but they do not themselves change, like what happens in a chemical reaction, catalyst the speed of the reaction, but they do not take part in the reaction; μ is the membrane structure which we have seen earlier with an outer most membrane called this in Skin membrane and inside membranes.

And the membranes have rules, suppose there are n membranes each membrane has rules, membrane 1 will have rules R_1 , membrane n will have rules R_n and so on. Small row 1, small row up to small row 1 they represents some sort of priority relation, that is when you have one rule and having higher priority than the other rule, the second rule cannot be used as long as you are able to apply the first rule, so some sort of a partial order is there among the rules and that is called the priority relation. And initially each membrane will contain some objects, they are known as W_1, W_2, W_n . So, the objects are arrange as string and represented, but the order of the symbols in the string does not matter as long as you consider symbol objects.

And one of membrane is taken as the output membrane and when the computation stops, whatever happens in the output membrane or whatever is left in the output membrane is taken as the output. Sometimes objects are sent out to the environment and that is taken as the output, both models are considered.

(Refer Slide Time: 18:36)



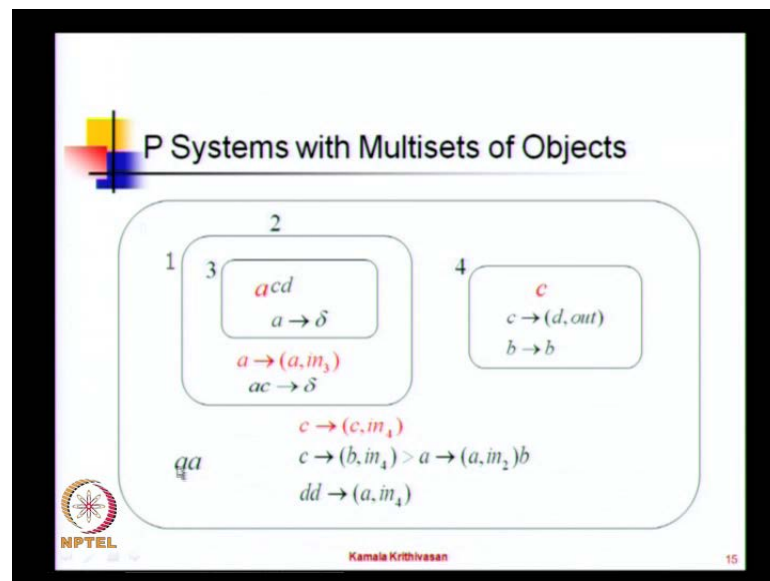
Now let us take an example and see what happens, this is an example with a multisets of objects or symbol objects, you have 1 membrane outside that is the skin membrane 1, within 1 you have 2, membrane 2 and within 2 you have membrane 3, 4 is another membrane which is within the membrane 1, but it is outside 2 and this is taken as the output membrane. The computation stops means you are not able to apply any rule, initially this contains two symbols c and d, this contains one symbol a, this contains three symbols a a c. The rules which can be applied within 1 membrane are given here, this is the rule which can be applied within 3, these are the two rules which can be applied in 2 and these are the rules which can be applied in 1 and these are the rules which can be applied in 4.

Initially a 4 does not have any other object, now you see that you have priority relation here, as long as there is the c you can use this rule or this rule for this c, but when you apply that, you cannot apply the rule a goes to a into b, for these two a. Only when you are not able to apply a rule for c you can apply a rule for a, this is called priority relation. Now let us see how this system behaves, everything happens parallel in all the membranes and the objects evolve parallelly. So, look at this, **this** delta means the membrane will dissolve, this object a will dissolve this membrane, similarly here if you have a and c in this membrane that will dissolve this membrane, delta represents the

solution. And here there are two symbols c and d , and the rule a goes to δ there is no a here, so no rule is applicable here, you have a here and no c then you cannot apply this rule, but you can apply the rule a goes to a and 3 here and the next instance this will be sent as the a inside membrane 3 . When an object evolves you also have something like in, out, etcetera, which tells you where the evolved object has to be sent.

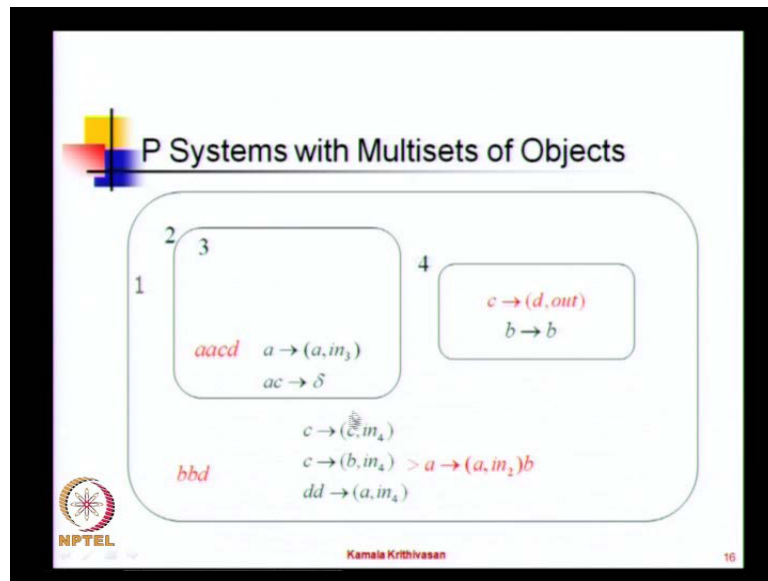
So, in 3 means a evolves as 3 and it is sent into this membrane, look at these three objects as I mentioned earlier you cannot apply this rule when you can apply this rule, so c evolves as b or c you can use the either of them. So, if you use this rule c evolves as c and it sent in membrane 4 .

(Refer Slide Time: 21:53)



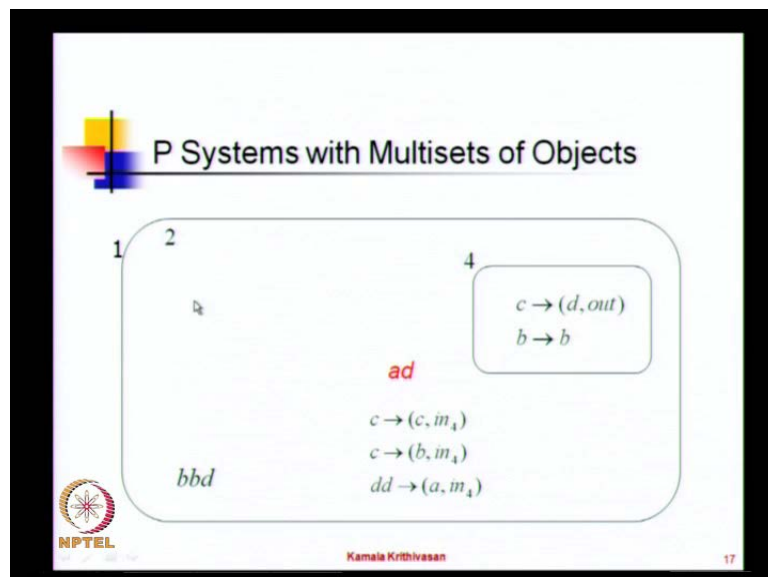
So, at the next step what we get is this, the a has been sent here c d remains as it is 2 a remain here, the c has been sent as c here. Now the next step you can make use of this rule, because there is a here and membrane 3 can get dissolved. Now there is no c here, so you can apply a rule for this a a each one of the a , one a will evolve into one a and kept here, one a will evolve into one a and one b , a will be sent into 2 **right**? Let us see, what happens?

(Refer Slide Time: 22:41)



So, in the previous step you had a because of applying this rule, this membrane gets dissolved c d comes out, here one a becomes one a and one b.

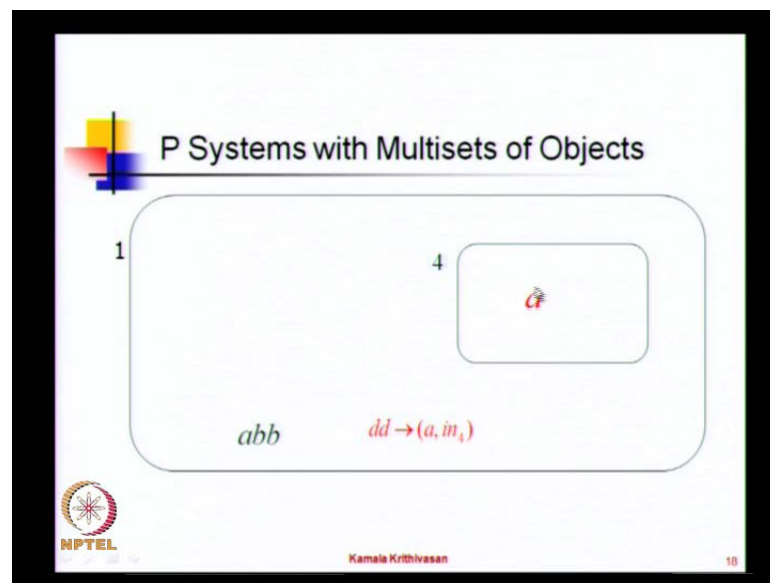
(Refer Slide Time: 23:52)



So, one a is sent here b remains here, so two a generate two a which are sent here and two b remain here, in this one there is the rule c is d out, so this c changes into d and this


sent out. So, in the next step we have a a c d here b b d here, because this membrane has dissolved, membrane 3 has dissolved 2 remains and within that you have this, 4 does not contain anything at this stage. Now you have one a and one c, so you can apply this rule, both the objects a c are consumed and the membrane is dissolved, you will be left with a and d and that comes out.

(Refer Slide Time: 24:19)




So, a d comes out you already have b b d here, this membrane has been dissolved. Now what happens you have a rule for d d, so this d and this d can combine and you can apply this rule both the d get consume and a is evolved and it is sent into 4. Now the situation is like this a has been sent here, a b b remains here, but I do not have any rule to apply here, so a b b remains here and then a is sent here, at this position the computation halts

(Refer Slide Time: 24:42)



Formal Definition (Contd...)

- ❖ The rules are of the following form:
 - > For symbol-objects: $u \rightarrow v$, where u is a string over V and $v = v'$ or $v = v' \delta$, where v' is a string over $\{a_{\text{here}}, a_{\text{out}}, a_{\text{in}} \mid a \in V\}$, and δ is a special symbol not in V ;
 - > For rewriting P systems: $X \rightarrow v(\text{tar})$, where $X \rightarrow v$ is a context-free rule and $\text{tar} \in \{\text{here}, \text{out}, \text{in}\}$;
 - > For splicing P systems: $(r, \text{tar}_1, \text{tar}_2)$, where $r = u_1 \# u_2 \$ u_3 \# u_4$ is a splicing rule over V and $\text{tar}_1, \text{tar}_2 \in \{\text{here}, \text{out}, \text{in}\}$;

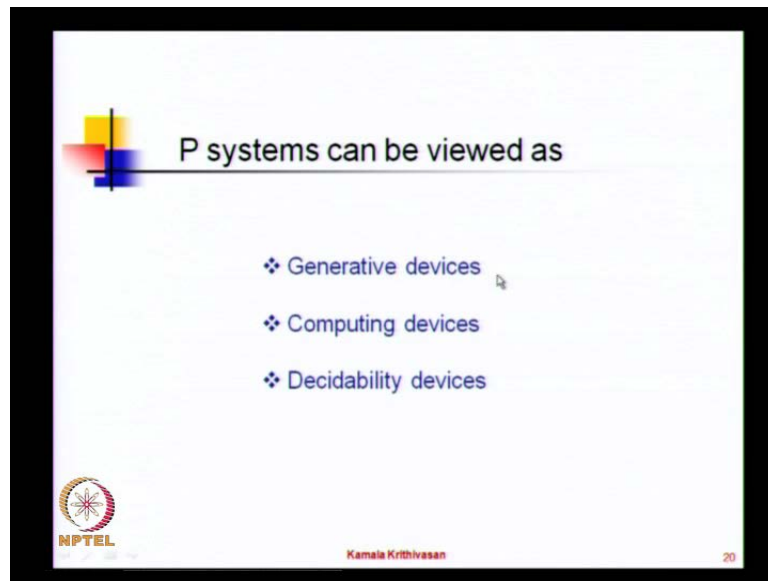


Kamala Krithivasan

10

So, we have the rules of this following form u goes to v , where u is a string over V and v is v' or $v' \delta$, **delta** means the membrane dissolves. Apart from that we also specify a target, a_{here} , a_{out} , a_{in} ; that is the object evolves as a and something will be sent into a membrane or sent out of the membrane or it will remain in the same membrane. Usually when it is a_{here} we do not specify it explicitly, if we do not specify a_{out} or a_{in} , it is understood that it remains in the same membrane. So, X goes to $V \text{tar}$, where x goes to v is a context free rule and tar will be here , out , here usually it is not specified.

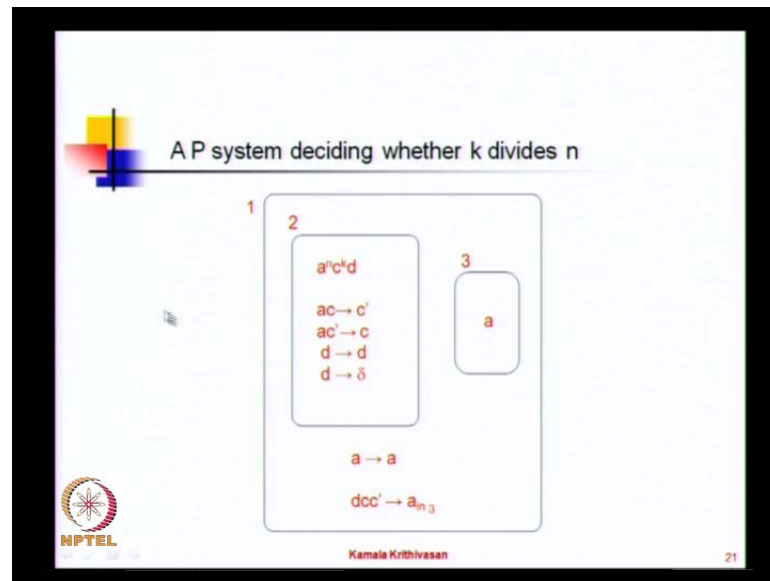
(Refer Slide Time: 25:37)



Now, you can look at the P systems as generative devices, generating some languages or computing devices, computing some functions or decidability devices, which tackle some decidability problems. It has been proved that they are computationally complete or you can simulate type 0 grammars given in the form of a matrix grammar. So, whatever you can do with a Turing machine, you can achieve with P system and so it is computationally complete system.

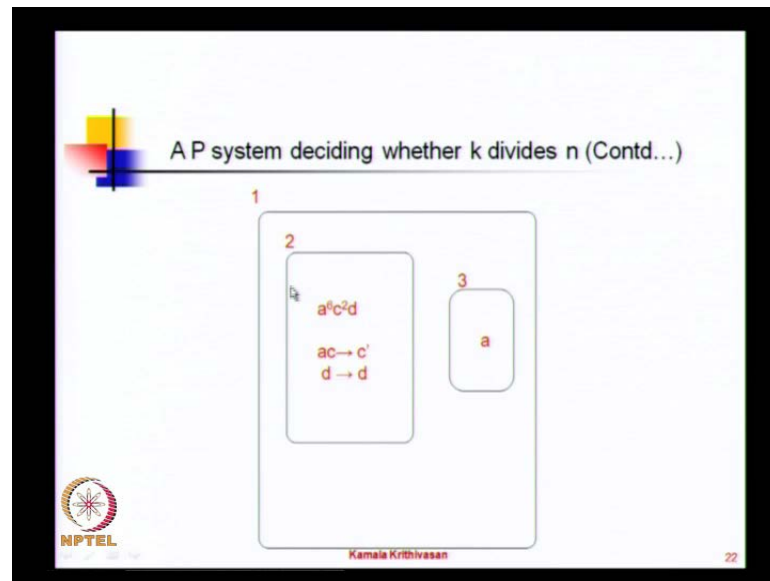
Now we must remember that whenever a new paradigm is defined, new computing paradigm is defined you must show that whatever is possible in the current paradigm, it is also possible with a new paradigm and so either you simulate with Turing machine or you register machine is new system can simulate a Turing machine or a register machine this you show or if you are not showing that directly indirectly you show that something equivalent to a Turing machine can be simulated. So, we will not go into those results at present we just give some example to show that how the membrane systems can be use for computing something or generating language or deciding some problem.

(Refer Slide Time: 27:16)



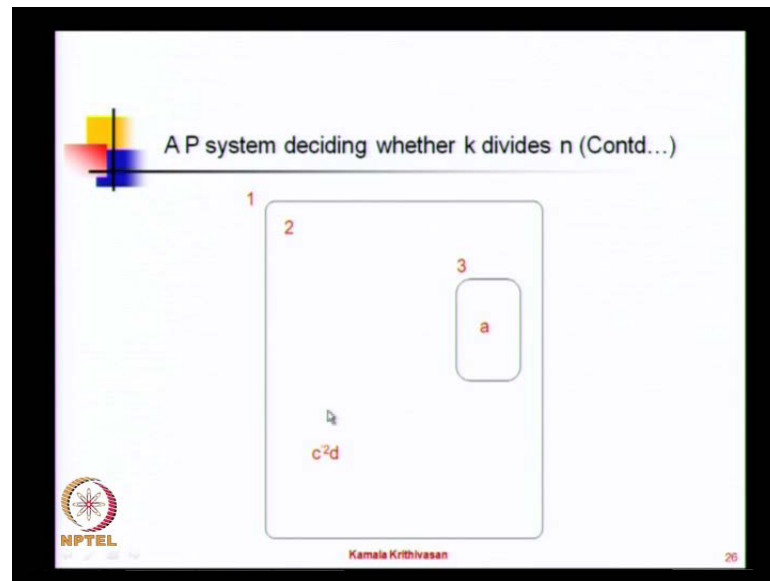
Let us look at this case we want to have a system which decides whether k divides n , k and n are two integers, you want to design a system which will find this out. So, we have this system, where in the outermost membrane or the Skin membrane is labeled 1, you have 2 membranes inside it 2 and 3, 2 has the following rules and 1 has this rule. And initially you load this with n a and k c , if you want to find out whether k divides n or not and 1 d . So, initially you start with objects n a k c and 1 d and at the end if you are left with 1 a here, that means, k divides n , if you are left with 2 a here that means, k does not divide n . So, when the computation stops you will have 1 a or 2 a here, 1 a will tell you that k divides n and 2 a will tell you that k does not divide.

(Refer Slide Time: 28:30)



Let us see how the system works taking some particular example, suppose I have 6 and 2, 2 divides 6, so I should get 1 a in the end here, let us see how this happens. We have a power 6 c squared d, using these rules d evolves as d only, we had all the 4 rules, but we may not be applying all of them simultaneously sometimes you will use something and otherwise when it is not applicable we will use another thing. So, with d it is possible to have d goes to d or d goes to delta, but if you use this the membrane will get dissolve that will not end up with anything. So, what happens is use the rule d goes to d and a c goes to c dash. So, 2 a and 2 c will be consumed to form 2 c dash. So, you will be left with 4 a and 2 c dash, now use the rule a c dash goes to c, this d is just used this rule and so it remains as d. So, 2 a and 2 c dash will be consumed to produce 2 c, so you end up with 2 a 2 c and 1 d, again repeating the same process using this rule 2 c dash will be produced.

(Refer Slide Time: 30:03)



So, you will end up with this, now we can use the rule $d \rightarrow \delta$, $d \delta$ and so this membrane will be dissolved and the entire thing comes out. Now I cannot apply any rule because the only rule applicable here is $c c \dashv d \rightarrow a$. So, that is not possible here, so the computation halts, when the computation halts I am left with 1 a here that means, k divides.

Suppose on the other hand I have a power 5 and c squared, 2 does not divide 5 in that case I should end up with 2 a here, let us see how this happens, again repeating the same process 2 a and 2 c are consumed to produce 2 c dash. So, we are left with this again 2 a and 2 c dash are consumed to produce 2 c. So, I will end up with this, at this stage only 1 a is there, so 1 a and 1 c will be consumed and c dash will be produced. So, we will end up with c c dash d at this time if you use the rule $d \rightarrow \delta$, the membrane 2 will be dissolved and it will come out and you have c c dash d here, there is a rule in membrane 1 $d c c \dashv \rightarrow a$ and 3. So, this rule is applicable and the a will go here there are no rules here, so this computation halts, producing 2 a. So, this shows that if at the end we end up with 1 a here k divides n, if we end up with 2 a k does not divide n, this is to show how something can be computed using the membrane system.


(Refer Slide Time: 31:47)

P system generating the Dyck set

$$\pi_4 = (\{a,b\}, \{a,b\}, \phi, [1]_1, a, (R_1, \phi), \infty),$$
$$R_1 = \{ a \rightarrow aa_{out}b, a \rightarrow a, b \rightarrow b, a \rightarrow a_{out}b, b \rightarrow b_{out} \},$$

a

a → aa_{out}b
a → a
b → b
a → a_{out}b
b → b_{out}

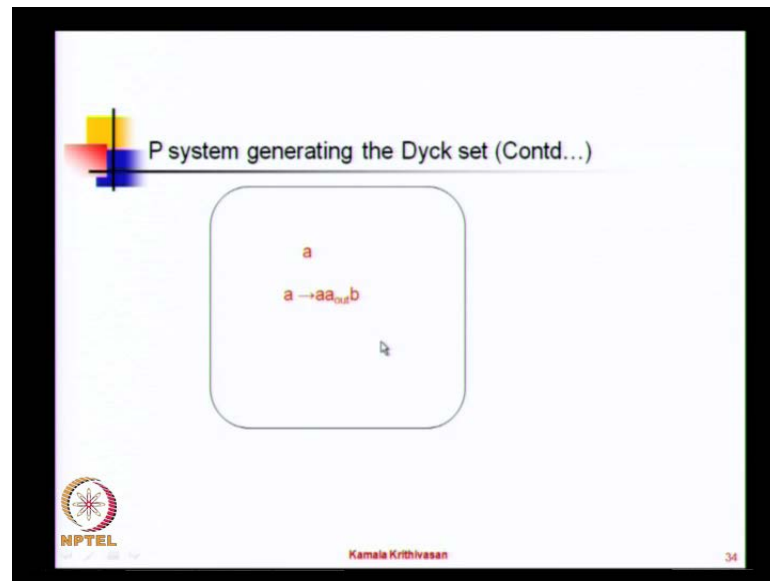
 NPTEL

Kamala Krithivasan

33

Now we can use it as a generative device, generating a language let us see how a membrane system can generate the Dyck set, what is the Dyck set? A Dyck set is the well formed strings of parenthesis, if you look at the left parenthesis as a, the right parenthesis as b, a string like a b is a well formed string, a a b b is a well formed string, a b a b is the well formed string and so on. This particular system with alphabet a b can generate, here again we are using it as a symbol object, let us see how the symbols are sent out, you have to take the order in which the symbols are sent out to the environment. There is no separate output membrane here, the environment collects the outputs and arranges them in the order in which they are sent out. So, you have only one symbol a here to start with and these are the rules. So, initially for this a if you use the rule a goes to a, a out b this a evolves into 2 a and 1 b out of which 1 is retained here, 1 is sent out.

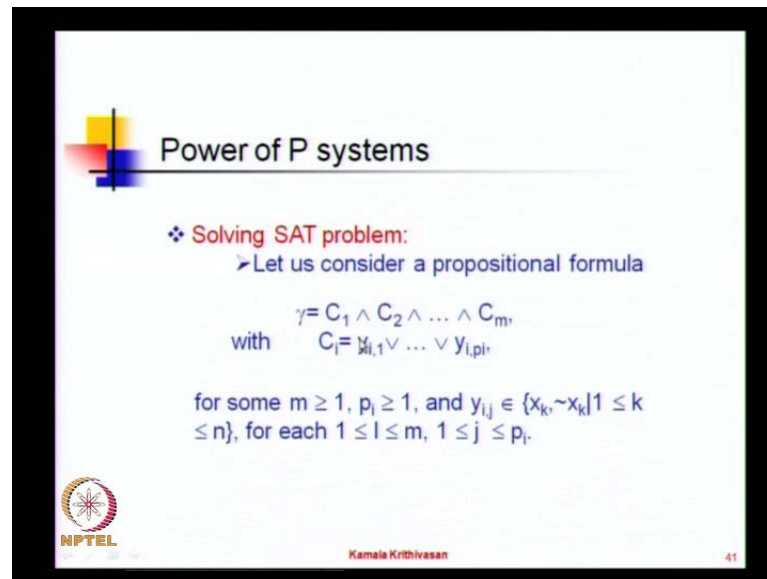
(Refer Slide Time: 33:05)



So, with this a if you apply this rule, you are left with 1 a and 1 b here and 1 a is sent out, now again for this a and b we can use the rule for b. if we use the rule b goes to b out it remains here for a again I can use the same rule a goes to a out b. So, I will be left with 1 more a and b and this a will be sent out. So, 2 a have been sent out now you have to collect the symbols in the order in which they come out, if two symbols come out at the same time they can be taken in any order, actually you have to consider both possibilities. So, inside we have a b b, b evolves, b can use b goes b or b goes to b out. So, if you use the rule b goes to out b comes out again this b also comes out you are left with a. So, for this a if you use the rule a goes to a out b, this evolves into 1 a and 1 b, b remains here a comes out a has come out b remains here, now if you use the rule b goes to b out, b comes out.

Now we have applied the rule in a particular order and we get this, **this** is the well formed strings of parenthesis a a b b a b, if you look at a as the left parenthesis, b as the right parenthesis, this is the well formed string and such a set is called a Dyck set, this we already know. If we have used the rules in a different order some of the string would have been formed here, but that will still be a well formed string of parenthesis, this can be checked very easily. And this example shows that you can use the membrane system or P system for a generating language like the Dyck set here.

(Refer Slide Time: 35:03)




Power of P systems

❖ **Solving SAT problem:**
➤ Let us consider a propositional formula

$$\gamma = C_1 \wedge C_2 \wedge \dots \wedge C_m,$$

with $C_i = \bigwedge_{j=1}^{p_i} y_{i,j}$

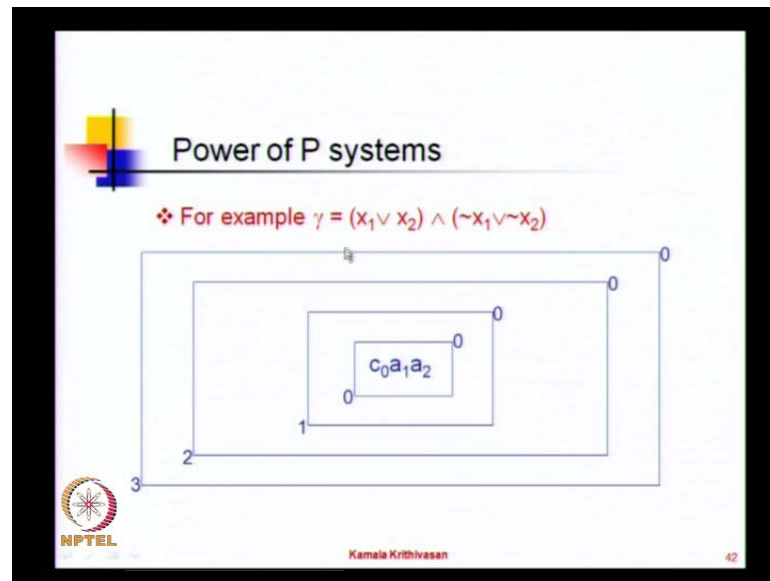
for some $m \geq 1$, $p_i \geq 1$, and $y_{i,j} \in \{x_k, \sim x_k \mid 1 \leq k \leq n\}$, for each $1 \leq i \leq m$, $1 \leq j \leq p_i$.

 NPTEL

Kamala Krithivasan 41

Now, you can also use the system to solve NP complete problems, that is the motivation really for using this unconventional models of computing. Let us consider the sat problem and see how this can be solved using a membrane system, here we are using a slightly different version of the membrane system, where the membranes have some charges they can acquire positive charge, negative charge or neutral charge. After the first model was defined there have been several variations and there have been more than 200, 300 papers in this area in the last 10 years. And several p h t c and m s this is have been written in this area and all the details are given in the P systems webpage, which can be easily accessed. Now let us see what is sat problem? We already studied this, we know that a Boolean expression can be expressed in conjunctive normal form as a conjunction of clauses c_1, c_2, c_m , are clause and they each c_i is of this form $Y_{i,1}, Y_{i,p_i}$, where each $Y_{i,j}$ is a literal. A literal is a variable or the negation of a variable, so it is of this form as an example you can consider this.

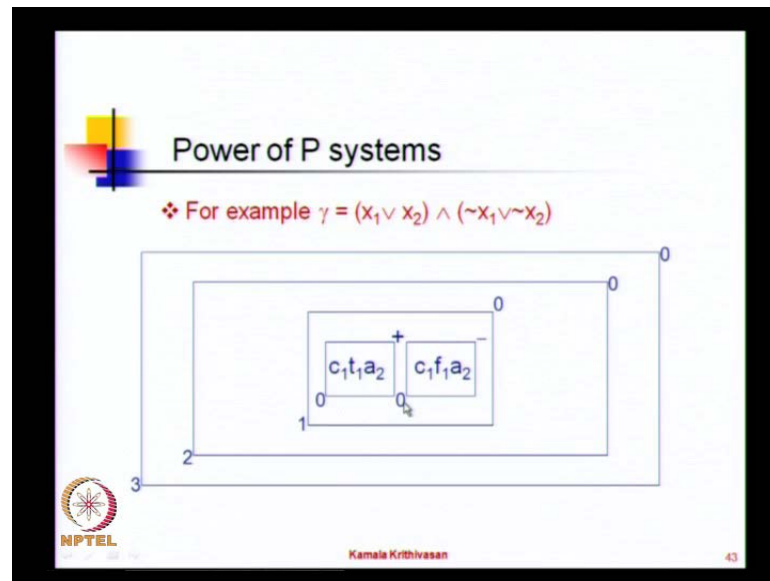
(Refer Slide Time: 36:35)



We want to design a P system, which we will find out whether this Boolean expression is satisfiable or not, this has got two clauses, this is the one, this is the one it has got two variables. The system for that will have two symbols a_1 and a_2 corresponding to the two variables X_1 and X_2 and one more symbol, which will keep on increasing c naught, c_1 , c_2 , which is a counter, which keeps track of the time. There will be $m + 2$ membranes one inside the other, where m is the number of clauses, there are two clauses here corresponding to them you have 2 membranes one innermost membrane 0, which contains all the symbols initially.

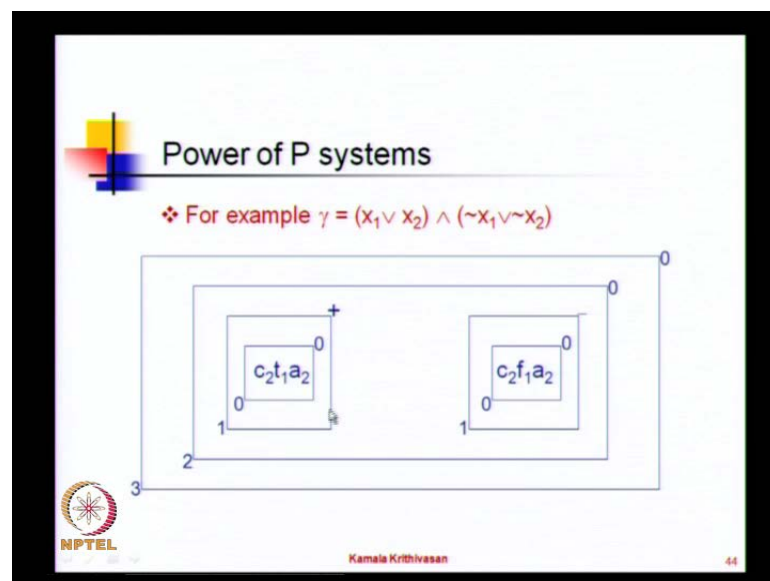
And outermost Skin membrane is there these symbols denote the labels of the membrane and what are these symbols? These symbols represent the charge, membrane have Neutral charge and initially a_1 can take the value T_1 or T_2 and that possibility is considered and you split it up as T_1 and f_1 consider splitting of a membrane.

(Refer Slide Time: 37:53)



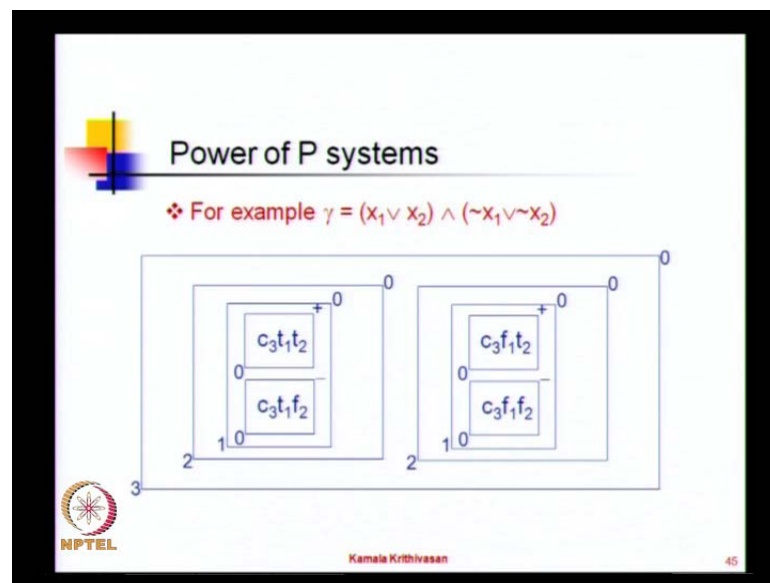
So, the membrane splits as 2 membranes the counter into is from c 0 to c 1 showing that one instance has elapsed, there are two possibilities T 1 and f 1, which has given here the membrane 0 splits into 2 membranes with labeled 0, but one gets a positive charge another one gets a negative charge. The next instance this positive charge is carried on to the next membrane and that membrane also is split into 2 membranes.

(Refer Slide Time: 38:37)



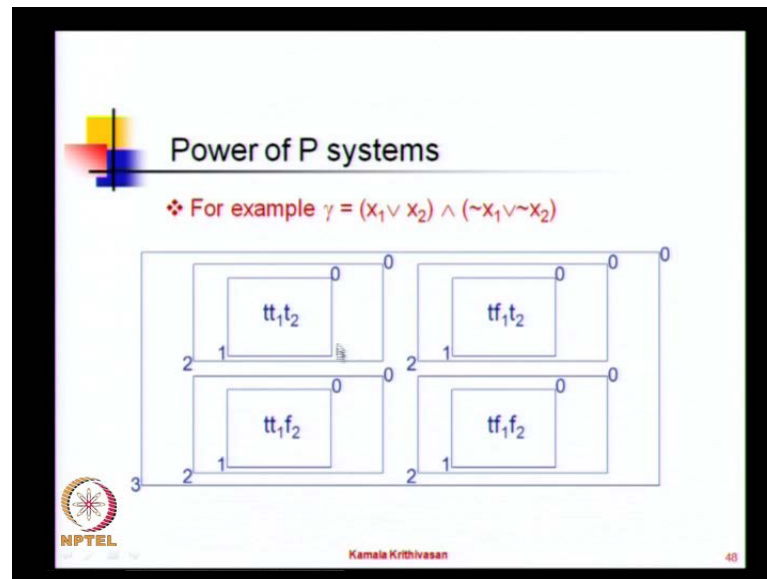
So, this one becomes a neutral charge whereas, the next membrane which has been split in the membrane 1, both the labels are 1. Please note then this gets a positive charge, this gets the negative charge. When the Neutral membrane is obtained the second variable is given the both the possible values T 2 and f 2 and again splitting takes place. So, you have T 2 and f 2 here, T 2 and f 2 here.

(Refer Slide Time: 38:59)



So, the four possible assignments are given by this, they acquire positive and negative charge then this membrane acquires Neutral charge. Now this positive charge is carried on here and this becomes Neutral and the next step everything becomes Neutral at that time, the time instance is $2n + 1c5$, now that $c5$ evolves into a particular symbol t .

(Refer Slide Time: 39:34)



Now what happens is whether this portion will be satisfied, there are two clauses, this is one clause will be satisfied if X_1 is a 1 or X_2 is a 1, it will not be satisfied if both are false. So, the next instance if some assignment satisfies the first clause the membrane dissolves, so you find that this assignment satisfies this assignment and so these 3 membranes will dissolve, but this will not get dissolve. So, you get this and the next instance you find out whether this assignment is or this clause is satisfied, this will be satisfied when both $X_1 X_2$ are false or one of them is false it will not be satisfied.

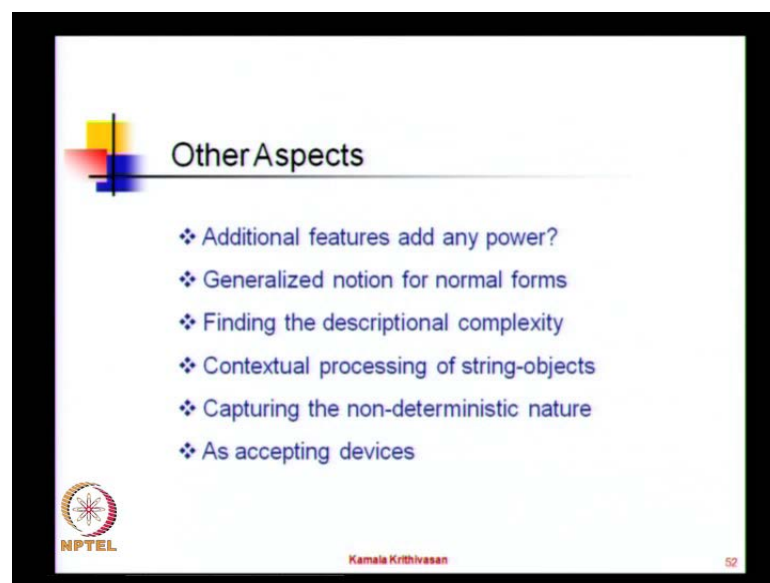
When both of them are true if it is satisfied the membrane 2 dissolves. So, you find that this dissolves right because it does not satisfy the first clause, this does not dissolve because it does not satisfies the second clause, this does not dissolve.

Now looking at this the next instance if there is a T it is ejected out of the membrane 3 which is the Skin membrane. So, next instance T is ejected out and you get 2 T, outside these 2 T tell you that there are two possible assignments which satisfies this Boolean expression in this particular model, I have not defined very formally I only informally described how it works.

And you see that the time taken is something like you know $2^n + 1$ plus say some m

or something like that where m is the number of clauses, here you can see that if X_1 is the 0 or X_2 is a 1 or X_1 is a 1 and X_2 is a 0 it will be satisfied, if both are 1 or both are 0 it will not be satisfied. So, out of the four assignments two assignment satisfies this, that is why you get 2 t outside the skin membrane. So, this just to show that you can solve NP-complete problem using a P system in linear time, similarly other problems like k dimensional matching Hamiltonian path problem they can also be solved in linear time.

(Refer Slide Time: 42:30)

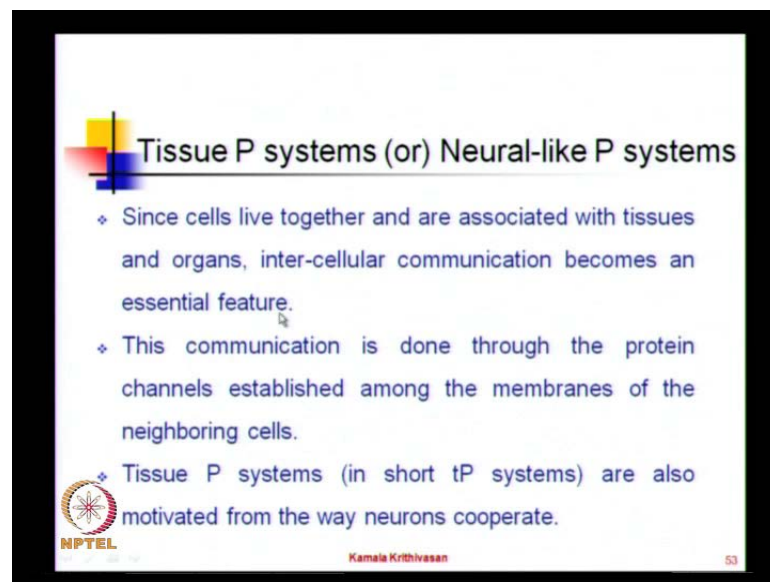


In the last 1 decade there has been lot of papers in this area of membrane computing, there have been additional features attached to the basic model, like you know an object can create a membrane and message carriers. How the objects carries messages they can have some positive charge, negative charge and some threshold value depending upon which the evolution takes place and so on. So, if you give additional features how does the system behave? This has been studied in detail and in all cases with some amount of membranes we have been able to achieve universal computable T, that is whatever you can achieve with the Turing machine you can achieve with this.

And it has been of interest to study with what it has been of interest to study the minimum number of membranes or the minimum number of rules used that is called Discretional complexity, there is be a trade of if you retry to reduce the number of




membranes the rules may increase and vice a versa. And also if you process string objects again splicing rules can be used and you can also have contextual rules for string objects, where contextual grammar is a particular type of a grammar and the nondeterministic use, how do you capture you can also design P Automata? Looking at it accepting device, these are some of the features which are consider in membrane computing, there have been other features as well this is only a few aspects, I am mentioning.

(Refer Slide Time: 44:34)



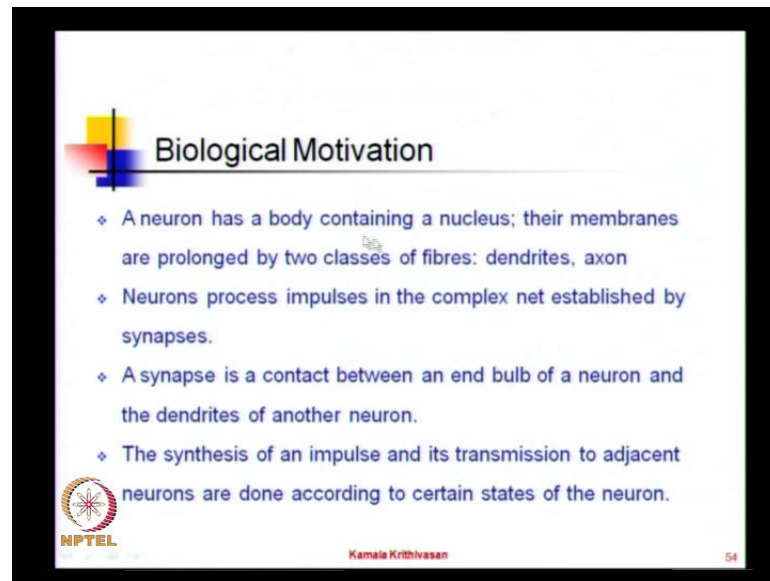
Tissue P systems (or) Neural-like P systems

- ◇ Since cells live together and are associated with tissues and organs, inter-cellular communication becomes an essential feature.
- ◇ This communication is done through the protein channels established among the membranes of the neighboring cells.
- ◇ Tissue P systems (in short tP systems) are also motivated from the way neurons cooperate.

 NPTEL  Kamala Krithivasan  53

Another type of P system, which has been studied is, tissue P systems and the tissue P systems they are more Neural like P systems, they resemble Neurons in particular manners cells are one particular type and neural cells or neurons are another type of cells and tissue P systems model them. These cells live together and are associated with tissues and organs intercellular communication becomes an essential feature, this communication is done through protein channels established among the membranes of the neighboring cells. So, these tissue P systems are motivated by the way Neurons cooperate.

(Refer Slide Time: 45:27)



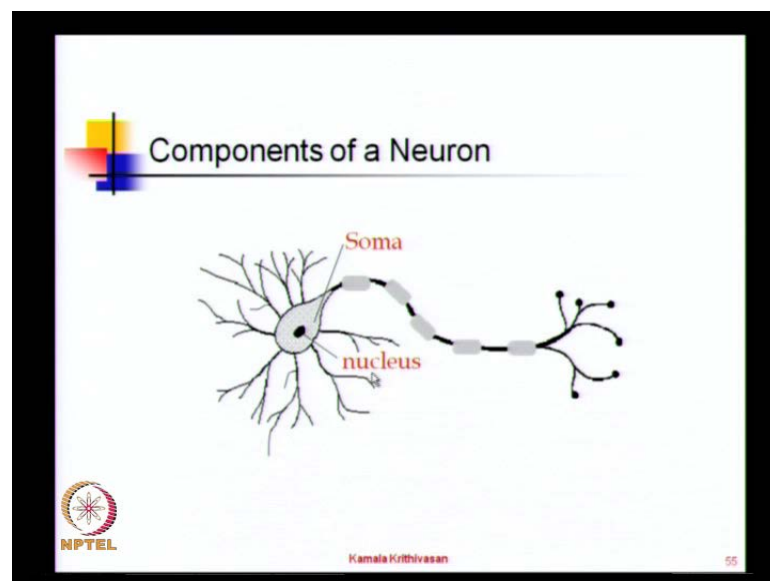
Biological Motivation

- ❖ A neuron has a body containing a nucleus; their membranes are prolonged by two classes of fibres: dendrites, axon
- ❖ Neurons process impulses in the complex net established by synapses.
- ❖ A synapse is a contact between an end bulb of a neuron and the dendrites of another neuron.
- ❖ The synthesis of an impulse and its transmission to adjacent neurons are done according to certain states of the neuron.

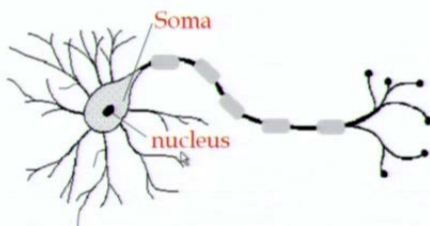
NPTEL Kamala Krithivasan 54

Neuron has a body containing a nucleus, their membranes are prolonged by two classes, Fibers Dendrites Axon and they process impulses in the complex net established by synapses, **synapses** is a connect between n bulb of a Neuron and a Dendrites of another Neuron, the synthesis of an impulse and it is transmission to adjacent Neurons are done according to the states of the Neuron.

(Refer Slide Time: 45:57)



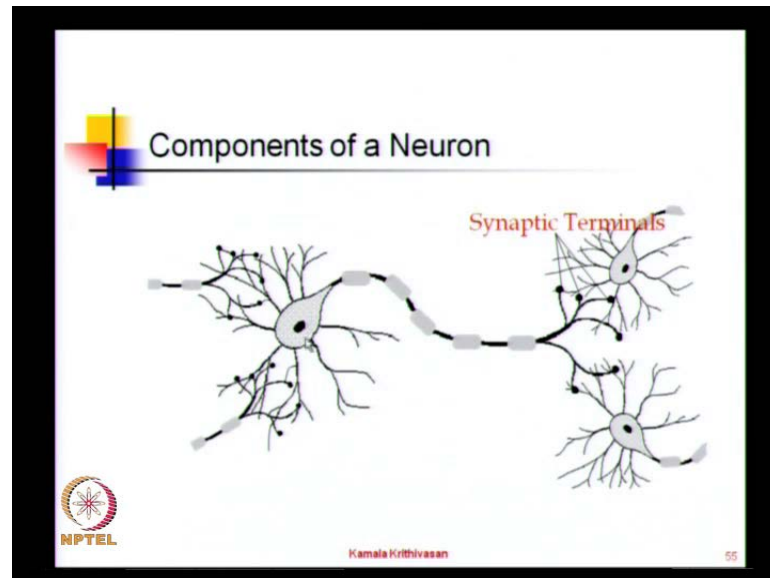
Components of a Neuron



NPTEL Kamala Krithivasan 55

So, here we also consider sheats look at this, this is a Neuron and these are the Dendrites of the Neuron this is as called Soma or the nucleus this is the Axon Myelin Sheaths.

(Refer Slide Time: 46:17)



So, this is one Neuron and this is another Neuron, this is another neuron and this sort of a thing connection between this and this they are called Synapses.

(Refer Slide Time: 46:32)

Definition: Neural-like P systems

♦ tP- systems is a construct

$$\pi = (O, T, \sigma_1, \sigma_2, \dots, \sigma_n, \text{syn}, i_{\text{out}})$$

where

$\text{syn} \subseteq \{1, \dots, m\} \times \{1, \dots, m\}$.

$\sigma_i = (Q_i, s_{i,0}, w_{i,0}, R_i)$

R_i is a finite set of rules of the form $sx \rightarrow s'y$ or $sx \rightarrow s'$ (y, tar) where $s, s' \in Q_i, x, y \in O^*$, $\text{tar} \in \{\text{go}, \text{out}\}$.

The NPTEL logo is in the bottom left, and 'Kamala Krithivasan' and '56' are in the bottom right.

So, in a formal way you can define it this way, but instead of going into the formal (()) definition, let me take an example and explain.

(Refer Slide Time: 46:42)

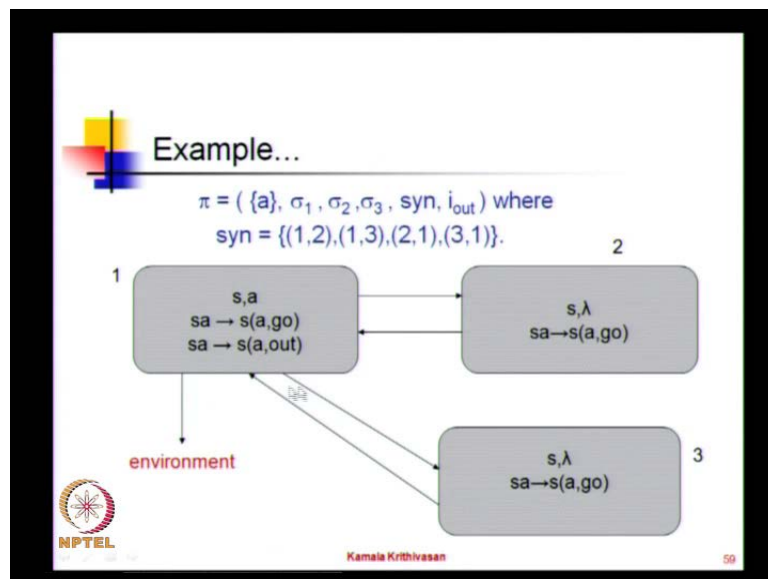
Transmitting the symbols

- Three transmitting modes are possible:
 - replication:** each symbol a , for (a,go) appearing in w' , is sent to each of the cells σ_j such that $(i,j) \in \text{syn}$;
 - one:** all symbols a appearing in w' in the form (a,go) are sent to one of the cells σ_j such that $(i,j) \in \text{syn}$, nondeterministically chosen;
 - spread:** the symbols a appearing in w' in the form (a,go) are nondeterministically distributed among the cells σ_j such that $(i,j) \in \text{syn}$.

NPTESyn. Kamala Krithivasan 59

You have something like this we have one Neuron, another Neuron, another Neuron, this is a cell.

(Refer Slide Time: 46:43)



Now these are Synapses, there is a direct connection between this and this and this, **this** and this, **this** and this, **this** is sending out to the environment, this is the output Neuron which sends things to the **output** environment. And with each Neuron a state is associated symbol objects are associated. Initially you have state S here, in this particular example there is only one state in general. You can have more than one state, each Neuron has a collection of states there is an initial state, initial set of symbols with that rules are there and the rules will be of this form one state and a string of objects, it is the number of objects represent as a string, this goes to another state and evolves into this, **this** is an set of objects x, they evolve as y

Here we consider three models where you can just use one rule or there are two rules we can use only one rule here or you can try to use one rule several times or two or three different rules for different symbol objects, they are called the one mode, the parallel mode and the maximal mode and **replication**- three ways, you can send the objects one is replicated mode that is for example, this a when it evolves into a, one copy will be sent here one copy will be sent here, that is the replication mode. In another mode this rule is used then s a goes to s and a and that a will be sent to only one of them, suppose an a evolves into 1 b and c then if you send both b and c to 1 membrane it is called the 1 mode; if you send one to one membrane, another to another membrane that is called the Spread mode.

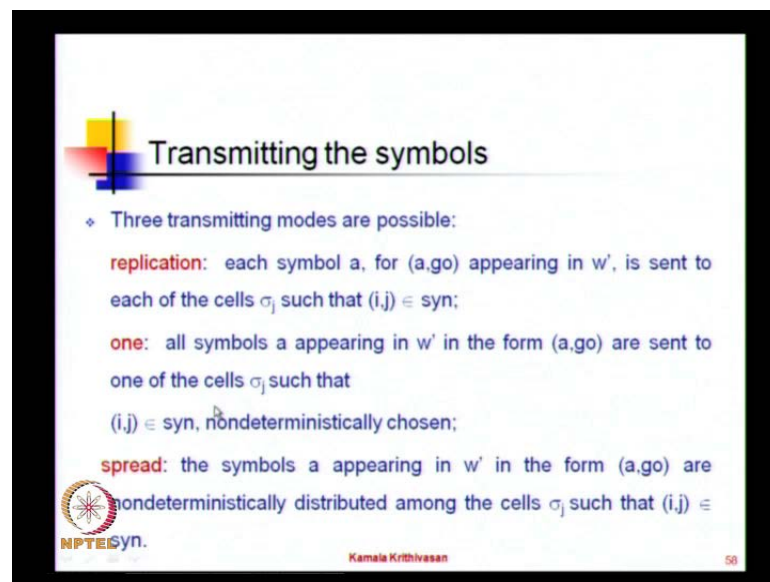
So, we have three types of transmitting modes replication in the replication, each symbol a is sent to each one of the cells σ_j , such that i, j belongs to Synapses that is if there is a connection between cell i and cell j then whatever evolves in cell i is sent to all. such a 1 all symbols a appearing in w dash or all symbols which have been evolved they are sent to the cells only one of the cells j and in the spread mode the symbols a appearing in w dash they are **nondeterministically** distributed among the cells σ_j .

So, in this example let us see what happens when you use the Minimal mode, when you use the Parallel mode, when you use the Maximal mode and so on. Suppose I use the parallel mode and send this out, so if you use the parallel mode a will evolve as a there is only one a here, so it will evolve as a. And if you use the mode replication for transmission, if you use the replicated mode this a will be sent as 1 a here and 1 a here,

then here you can use this rule here also.

So, this will evolve into another a and will be sent back. So, at the third step you get 2 a here, because the rule replication mode is parallel for the 2 a, you have to apply the same rule. So, when I apply this rule 2 a will be sent here, 2 a will be sent here and again using these rules those 2 a will be sent back and these 2 a will be back. So, you will end up with 4 a, at the 5 instance of time, at the 7 instance you will end up with 8 a and so on.

(Refer Slide Time: 51:30)



Transmitting the symbols

- ♦ Three transmitting modes are possible:
 - replication:** each symbol a, for (a,go) appearing in w', is sent to each of the cells σ_j such that $(i,j) \in \text{syn}$;
 - one:** all symbols a appearing in w' in the form (a,go) are sent to one of the cells σ_j such that $(i,j) \in \text{syn}$, nondeterministically chosen;
 - spread:** the symbols a appearing in w' in the form (a,go) are nondeterministically distributed among the cells σ_j such that $(i,j) \in \text{syn}$.

NPTESyn. Kamala Krithivasan 50

So, once you start emitting using this rule, all the a will be simultaneously sent out and you will get 2 power n a, that is what is meant by this n parallel replication is 2 power n, if you use the Minimal mode.

(Refer Slide Time: 51:32)

Example...

- ❖ $N_{\min, \text{rep}}(\pi_1) = \{ (n) \mid n \geq 1 \},$
- ❖ $N_{\min, b}(\pi_1) = \{ (1) \},$ for $b \in \{\text{one, spread}\},$
- ❖ $N_{\text{par}, \text{rep}}(\pi_1) = \{ (2^n) \mid n \geq 0 \},$
- ❖ $N_{\text{par}, b}(\pi_1) = \{ (1) \},$ for $b \in \{\text{one, spread}\},$
- ❖ $N_{\max, \text{rep}}(\pi_1) = \{ (n) \mid n \geq 1 \},$
- ❖ $N_{\max, b}(\pi_1) = \{ (1) \},$ for $b \in \{\text{one, spread}\}.$

NPTEL
Kamala Krithivasan
60

And 1 mode this a will go here, again go here, will go here, will go here and so on. Finally, at some time it will be sent out, so, only 1 a will come out, so what you get is just 1. So, similarly the other things are presented here again this has lot of a application in the sense that, this can be used in different ways can be looked at as variant model of the membrane system.

(Refer Slide Time: 52:19)

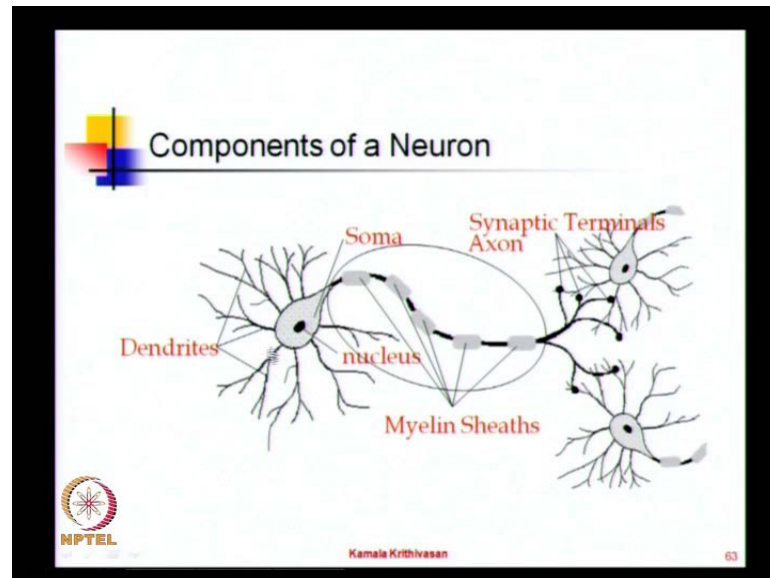
Introduction - SN P Systems

- › SN P systems is a computational model that has been inspired by neurobiology
- › **some ideas incorporated are**
 - › synapses, with the replication of impulses in the case of multiple synapses,
 - › linking a neuron to several neighboring neurons
 - › state of a neuron.
- › aspect captured is the fact that most of the neural impulses are almost identical, electrical signals of a given voltage, with a crucial role played by the time when these signals are issued, hence by the intervals between signals.

NPTEL
Kamala Krithivasan
62

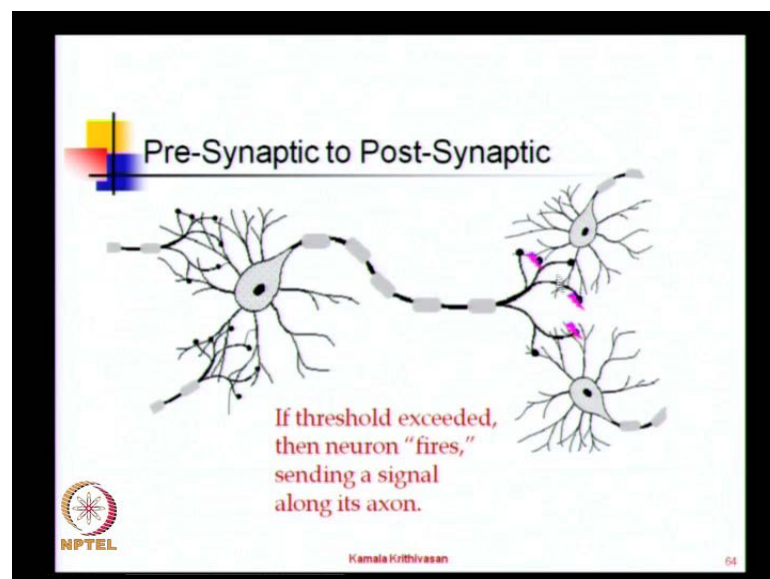
Recently another sort of membrane model has been defined this is called SNP systems or Spiking Neural P system, which more or less resembles the Neuron in an exact manner.

(Refer Slide Time: 52:27)



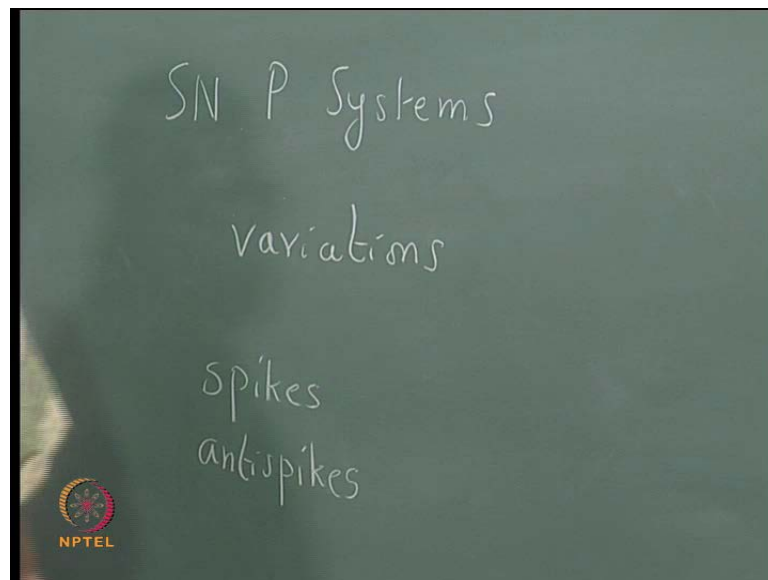
You have Spikes in a Neuron and they spike, and then go through to another membrane, this is the situation which represents the Neurons in the connection between them.

(Refer Slide Time: 52:45)



What happens is you have some spikes, and these spikes when they enter the Neuron, like this when they exceed the threshold value, they spike again this neuron spikes and the spike is sent out. So, this comes out and splits and gets into the Synapses here and this can be model in a very efficient manner and we get what is known as Spiking neural P systems. The current interest is in this spiking neural P systems, because they seem to have a lot of similarity between the spiking neural P systems and Petri nets, Petri nets is a very well known model and so, you can use it for many practical purposes.

(Refer Slide Time: 53:51)



So, you have Petri nets in the one hand and on the other hand you have S N P systems, here again there are several variation or there are mild difference in the definition. And in Petri nets also you have variations, you have colored Petri nets, timed Petri nets and so on. So, it is an interesting study to study what is possible here, how to simulate it with the corresponding model, so one model will correspond with some particular model, here another way a particular model can simulate this, which one can simulate within an efficient manner, that is an interesting model of study.

And in SNP system we have seen only spikes we can also have anti-spikes, and that is the current interest. Now how we can use spikes and anti-spikes to generate binary strings, so that a language will be generated. What sort of a language will be generated

by a S N system or a SNP system and so on? So, that is an interesting area of study, so this model of paradigm membrane computing has several variations and is a current area of research, even though initially it will be useful to simulate it with actual cells, later it was found that it is not possible. So, currently the interest is how to make use of this interesting paradigm and use it in the conventional model simulate and get good results? So that we can solve some difficult problems in an efficient manner.