1/

Hello, everyone, This is group 26. We will have a presentation about TLA+ and TLA+ toolbox.

2/

we will first introduce the basic information of the TLA+ and TLA+ toolbox.

Next the core conceptions and important ideas about it.

Then the Grammar and the usage of TLA+ Toolbox.

Finally, there are two demos.

3/

The first section is a basic introduction.

4/

We have learned Petri-Net before. Petri net is one of several [mathematical](https://en.wikipedia.org/wiki/Mathematical) [modeling languages](https://en.wikipedia.org/wiki/Modeling_language) for the description of [distributed systems](https://en.wikipedia.org/wiki/Distributed_systems).

Similarly, TLA+ is a kind of formal specifical language developed by Leslie Lamport, a famous computer scientist.

TLA+ is used to design, document, and verify software, hardware, and real-life systems, especially distributed systems and concurrent systems.

TLA+ is based on mathematics and does not resemble any programming language.

5/

Here comes the TLA+ toolbox. TLA+ toolbox is an IDE help us to parse or generate printed-vision model, and integrated TLC checker.

You can also use VS code extensions of TLA+

6/

In the second section, we will introduce core conceptions and important idea about TLA+

7/

First of all, TLA+ is state-based. When describing a system, it models an execution of a system as **a sequence of states.**

And in many related resources, a sequence of states is called behavior. A pair of consecutive states is defined as a step.

A TLA+ program tries to describe a set of behaviors with two things:

One is initial condition, specified the initial state of a system,

another is next-state relation, specified possible action from a state.

Almost the same as describing a state machine.

From my perspective, when trying to model a system with TLA+, we define the set of ordered pairs by describing the next-state relation。

8/

After modeling the system, we need to check the property of a system.

In TLA+, we have two kinds of properties: Invariant and Temporal Formula.

An invariant is a property of a mathematical object that remains unchanged after operations or transformation

Temporal Formula describes temporal logic; temporal logic means it’s Boolean value may change in time.

Pay attention, although TLA+ is based on mathematics. Most times, it cannot check the property “theoretically.”

9/

That means TLA+ tries to find whether a property is satisfied in practical-size state space by searching each state.

By default, the searching method is BFS.

For example, in the state space shown,

it first checks the root

then the adjacent 3 states of level 2

and finally, the outer of level 3.

17/

Here comes the TLA+ Toolbox part, in this part we will show the usage of TLA+ Toolbox with a simple demo.

18/

You can download the Toolbox from this link, and after download it, open it, and creating a new spec by click File-Open Spec-Add new Spec, name it and your workspace is ready.

19/

Here is a Demo from Petri Net assignment;

we will use TLA+ to test whether it satisfies “for any reachable marking, token in p2 is less than or equal to the token in p3.”

and test whether marking m=(0,1,10) can happen infinitely often

20/

In the TLA+ editor, you can build your model here

 If model is parsed, a symbol will be shown in the bottom:

 That means your model is correct.

21/

First, we model the petri net with TLA+

the Variable: state defines the places in the Petri net, and constant will be defined as the set of state index

Init defines the initial state, in which except place 0 is 1 other place is empty.

Each operation is defined as below

for example, tB means,

when at the current state, if place 1, has more than 1 token,

then transition tB can be fired,

and in next state, place 3 add one token,

place 1 minus 1 token,

the total possible transition is defined as Next;

the expression means,

in a state, if the condition is met, tA or tB or tC or tD can happen

22/

For verification 2, we first describe the property that is shown in the Left, we will regard it as an invariant property, means for any state, token in place 2 is no more than place 3.

For verification 3,

we first divide it into two steps:

A: is from initial marking able to arrive at (0,1,10)?

B: is from (0,1,10) is able to come back (0,1,10)?

if both A and B is true, then the original proposition is true

Next, we modeled it by these two formulae,

the first means we can never arrive at (0,1,10)

the second means, the counter of times arriving at (0,1,10) can never become 2

23/

After building the model, you can click up bar of the window, and create a new checker model from here

24/

There are overall 1-5 steps to set and run this model checker

25/

Step 1 and 2 is to specify initial state and next-state relation

you need add your init state and next-state relation in this part

step 3 is to give value of the constants part in the specification

here we set the state number the set {1,2,3}

26/

But remember, we have mentioned that,

TLA+ is not able to test “theoretically.” , we need to smaller the state space to accelerate the search process

So for testing, we set the upper bound of states 2 and 3.

27/

For verification 2, after pushing check button, the toolbox finds an error,

that is, the invariant is not hold, and it finds a trace.

28/

Then doing the same thing for verification 3

we tick on q3\_1, the result is not hold

we tick on q3\_2, the result holds.

So A hold while B not hold, so the original proposition is false、

29/

In this section, we will show another demo which also based on Petri Net Assignment

30/

Here is the Demo 2, we check whether there is a deadlock

31/

Here is the result page

32/

The program is too long to show here,

after checking, we can find an error,

which means there is a deadlock,

and TLA+ gives the trace.

33/

There are also other more complicated application of TLA+, for example, we also finished a TLA+ model of Afek & Gafni election algorithm, and lamport himself verified his Paxos Algorithm with TLA+

Here is an overview of today’s presentation.

First, we introduce the formal specifical language TLA+,

which is based on mathematical

and is a top-layer description of systems.

Second, we introduce the state machine, property check process,

Third, we introduce the language grammar and usage of the toolbox

Finally, two demos are given here to help us understand it.

34/

Here are several extra resources,

the official website,

a third-party introduction,

official video resources,

and a Google discussion group.

The only important suggestion is,

DO practice using it by yourselves.

It is quite hard to learn it just by going through books or videos.