**CCSLRLSynthesizer Documentation**

**Submitter : Ming Hu**

[**ecnu\_hm@163.com**](mailto:ecnu_hm@163.com)

**East China Normal University**

1. **Environment setting requirements**

* Python 2.7

If you use ubuntu system, you can install python by using command:

*apt-get install python-pip*

* Numpy

You can install numpy by using command:

*pip install numpy*

**We provide a virtual machine image containing all running environments and artifact.**

**You can download VM image from two sources.**

1. **Google Drive (Link:** [**https://drive.google.com/file/d/1-03Am76jD6BDDeBVHBzDWhherZwpU\_Wc/view?usp=sharing**](https://drive.google.com/file/d/1-03Am76jD6BDDeBVHBzDWhherZwpU_Wc/view?usp=sharing)**).**
2. **Baidu Cloud Disk (Link：**[***https://pan.baidu.com/s/1eyv6Qguh570U-8MTT5IAZA***](https://pan.baidu.com/s/1eyv6Qguh570U-8MTT5IAZA) **Extraction Code：*RTSS*).**

**Note that, the VM image is generated by using VirtualBox (**[**https://www.virtualbox.org**](https://www.virtualbox.org/)**).**

**The password of VM is *rtss\_162* .**

1. **Code Structure**

**--CCSL**

**|--\_\_init\_\_.py**

**--CCSLChecker**

**|--\_\_init\_\_.py**

**--CCSLPaser**

**|--\_\_init\_\_.py**

**--CCSLRLSynthsizer.py**

**--main.py**

* **CCSL** defines the data structure of CCSL specification and its elements relation, expression, clock, hole respectively.
* **CCSLChecker** defines the functions of deduction
* **CCSLPaser** defines the functions to deal with the input from files
* **CCSLRLSynthsizer.py** implements a RL algorithm to archive CCSL synthesis.
* **main.py** is he entrance of the program

1. **Instruction**

**3.1 Command**

**Step 1:**

*cd /CCSLRLSynthesizer*

**Step 2:**

(If you want to **synthesize a CCSL specification**, you can run **command (1)**. If you want **test the experiment results from paper**, you can run **command (2)**)

1. **Synthesis a CCSL specification**

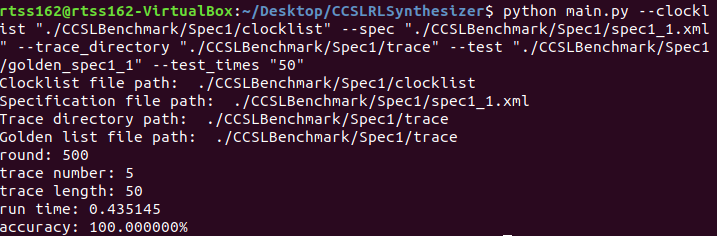
*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_2.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec2/trace"*



Using this command, we can get the complete CCSL specification from incomplete CCSL specification in file *Spec2\_2.xml*.

1. **Accuracy and time testing**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_1.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_1" --test\_times 50*



Using this command, we can test the probability that the synthesizer can generate the expected golden specification and average run time.

**(3) Annotations of command parameter:**

* **--clocklist:** The path of file clocklist which define all the clock used in specification.
* **--spec:** The path of specification file which define the incomplete specification.
* **--trace\_directory:** The path of trace directory which contains all the trace files.
* **--round:** Number of rounds of reinforce learning training
* **--trace\_length:** The length of trace used in reinforcement learning training.
* **--trace\_number:** The number of trace used in reinforcement learning training.
* **--test:** The path of golden solution file. This parameter is used to check whether our method can generate the expected golden solution.
* **--test\_times:** The number of tests is used to estimate the probability that our algorithm can solve the golden solution.

#### 3.2 Input file

**3.2.1 Incomplete CCSL specification**

We use XML to describe CCSL specifications.

File structure is as follow:

<CCSLConfigure>  
 <relations>  
 <relation type='0' leftClock="c7" rightClock="e1"/>  
 …  
 </relations>  
 <expressions>  
 <expression name="e0" type='1' leftClock="c0" rightClock="c1" addition=""/>  
 …  
 </expressions>  
</CCSLConfigure>

1. **CCSLCondigure**

The root node of the xml file is ***CCSLConfigure***, in which contains node *relations* and *expressions*.

1. **Relation**

**The node *relations*** contains serval *relation* nodes which define relations of CCSL specification.

**The node** ***relation*** is shown in *<relation type=’type\_value’ leftClock=’clock\_name’ rightClock = ‘clock\_name’>* to define a relation in CCSL specification.

The ***type*** specifies the relation type.

|  |  |
| --- | --- |
| type | relation type |
| 0 | Coincidence |
| 1 | Precedence |
| 2 | Causality |
| 3 | Exclusion |
| 4 | Subclock |

Note that we can define a relation operator hole by setting “*type = ‘-1’*”

The ***leftClock*** and ***rightClock*** specify the relation clock.

We can define a relation clock hole by setting “*leftClock = ‘’*” or “*rightClock = ‘’*”

1. **Expression**

**The node *expressions*** contains serval *expression* nodes which define expressions of CCSL specification.

**The node** ***expression*** is shown in *<expression type=’type\_value’ leftClock=’clock\_name’ rightClock = ‘clock\_name’ addition=’addition\_value’>* to define a relation in CCSL specification.

The ***type*** specifies the expression type.

|  |  |
| --- | --- |
| type | relation type |
| 0 | Union |
| 1 | Intersection |
| 2 | Infimum |
| 3 | Supremum |
| 4 | Delay |
| 5 | Periodicity |

Note that we can define a expression operator hole by setting “*type = ‘-1’*”

The ***leftClock*** and ***rightClock*** specify the expression clock.

We can define a relation clock hole by setting “*leftClock = ‘’*” or “*rightClock = ‘’*”

The ***addition*** specifies the addition parameter in expression delay or perioficity.

* + 1. **clocklist**

We use this file to declare atomic clocks.

File structure is as follow:

*Clock clock\_name1*

*Clock clock\_name2*

* + 1. **trace**

We use this file to define expected traces.

File structure is as follow:

<trace:Trace name="newfile2018\_1113\_142123">  
 <logicalSteps nextStep="//@logicalSteps.1">  
 <eventOccurrences referedElement="//@references.0" eState="noTick" fState="noTick"/>  
 <eventOccurrences referedElement="//@references.1" counter="1"/>

…  
 </logicalSteps>

….

</trace>

The node **logicalSteps** specifies the clocks state (clock tick or not) in one logical step.

The node eventOccurrences specifies a clock state (clock tick or not) in current logical step.

The parameter ***referedElement*** specifies a clock reference id in CCSL specification.

The parameter ***counter*** specify the tick time of correspond clock in current logical step.

#### 3.3 Experiment Coverage in paper

**3.3.1 Test Commands of Table V:**

1. **Spec1<4,1,3>**

* **case1(0,0,3)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_1.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_1" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_1.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_1" --test\_times 500*

* **case2(0,1,2)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_2.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_2" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_2.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_2" --test\_times 500*

1. **Spec2<10,5,10>**

* **case1(0,0,10)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec2/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_1.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec2/trace" --test "./CCSLBenchmark/Spec2/golden\_spec2\_1" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec2/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_1.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec2/trace" --test "./CCSLBenchmark/Spec2/golden\_spec2\_1" --test\_times 500*

* **case2(4,2,4)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec2/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_2.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec2/trace" --test "./CCSLBenchmark/Spec2/golden\_spec2\_2" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec2/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_2.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec2/trace" --test "./CCSLBenchmark/Spec2/golden\_spec2\_2" --test\_times 500*

1. **Spec3<20,6,16>**

* **case1(9,1,7)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec3/clocklist" --spec "./CCSLBenchmark/Spec3/Spec3\_1.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec3/trace" --test "./CCSLBenchmark/Spec3/golden\_spec3\_1" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec3/clocklist" --spec "./CCSLBenchmark/Spec3/Spec3\_1.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec3/trace" --test "./CCSLBenchmark/Spec3/golden\_spec3\_1" --test\_times 500*

* **case2(5,1,10)**

**2000 round**

*python main.py –clocklist “./CCSLBenchmark/Spec3/clocklist” –spec “./CCSLBenchmark/Spec3/Spec3\_2.xml” –round 2000 –trace\_length 50 –trace\_number 5 –trace\_directory “./CCSLBenchmark/Spec3/trace” –test “./CCSLBenchmark/Spec3/golden\_spec3\_2” –test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec3/clocklist" --spec "./CCSLBenchmark/Spec3/Spec3\_2.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec3/trace" --test "./CCSLBenchmark/Spec3/golden\_spec3\_2" --test\_times 500*

1. **Spec4<9,4,5>**

* **case1(0,4,0)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec4/clocklist" --spec "./CCSLBenchmark/Spec4/Spec4\_1.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec4/trace" --test "./CCSLBenchmark/Spec4/golden\_spec4\_1" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec4/clocklist" --spec "./CCSLBenchmark/Spec4/Spec4\_1.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec4/trace" --test "./CCSLBenchmark/Spec4/golden\_spec4\_1" --test\_times 500*

* **case2(5,1,1)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec4/clocklist" --spec "./CCSLBenchmark/Spec4/Spec4\_2.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec4/trace" --test "./CCSLBenchmark/Spec4/golden\_spec4\_2" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec4/clocklist" --spec "./CCSLBenchmark/Spec4/Spec4\_2.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec4/trace" --test "./CCSLBenchmark/Spec4/golden\_spec4\_2" --test\_times 500*

1. **Spec5<25,74,54>**

* **case1(1,1,5)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec5/clocklist" --spec "./CCSLBenchmark/Spec5/Spec5\_1.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec5/trace" --test "./CCSLBenchmark/Spec5/golden\_spec5\_1" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec5/clocklist" --spec "./CCSLBenchmark/Spec5/Spec5\_1.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec5/trace" --test "./CCSLBenchmark/Spec5/golden\_spec5\_1" --test\_times 500*

* **case2(3,1,0)**

**2000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec5/clocklist" --spec "./CCSLBenchmark/Spec5/Spec5\_2.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec5/trace" --test "./CCSLBenchmark/Spec5/golden\_spec5\_2" --test\_times 500*

**5000 round**

*python main.py --clocklist "./CCSLBenchmark/Spec5/clocklist" --spec "./CCSLBenchmark/Spec5/Spec5\_2.xml" --round 5000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec5/trace" --test "./CCSLBenchmark/Spec5/golden\_spec5\_2" --test\_times 500*

* + 1. **Test Commands of Figure 4**

**We can complete the experiments in Figures 4, 5 and 6 by changing the values of parameters “***--round***”.**

* **Spec1(0,1,2)**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_2.xml" --round* ***5000//change here*** *--trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_2" --test\_times 500*

* **Spec2(4,2,4)**

*python main.py --clocklist "./CCSLBenchmark/Spec2/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_2.xml" --round* ***5000//change here*** *--trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec2/trace" --test "./CCSLBenchmark/Spec2/golden\_spec2\_2" --test\_times 500*

* **Spec3(5,1,10)**

*python main.py –clocklist “./CCSLBenchmark/Spec3/clocklist” –spec “./CCSLBenchmark/Spec3/Spec3\_2.xml” –round* ***5000//change here*** *–trace\_length 50 –trace\_number 5 –trace\_directory “./CCSLBenchmark/Spec3/trace” –test “./CCSLBenchmark/Spec3/golden\_spec3\_2” –test\_times 500*

* **Spec4(5,1,1)**

*python main.py --clocklist "./CCSLBenchmark/Spec4/clocklist" --spec "./CCSLBenchmark/Spec4/Spec4\_2.xml" --round* ***5000//change here*** *--trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec4/trace" --test "./CCSLBenchmark/Spec4/golden\_spec4\_2" --test\_times 500*

* **Spec5(1,1,5)**

*python main.py --clocklist "./CCSLBenchmark/Spec5/clocklist" --spec "./CCSLBenchmark/Spec5/Spec5\_1.xml" --round 2000 --trace\_length 50 --trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec5/trace" --test "./CCSLBenchmark/Spec5/golden\_spec5\_1" --test\_times 500*

* + 1. **Test Commands of Figure 5**

**We can complete the experiments in Figures 5 by changing the values of parameters “***--trace\_length***”.**

* **Spec1(0,1,2)**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_2.xml" --round 2000 --trace\_length* ***50//change here*** *--trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_2" --test\_times 500*

* **Spec2(4,2,4)**

*python main.py --clocklist "./CCSLBenchmark/Spec2/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_2.xml" --round 2000 --trace\_length* ***50//change here*** *--trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec2/trace" --test "./CCSLBenchmark/Spec2/golden\_spec2\_2" --test\_times 500*

* **Spec3(5,1,10)**

*python main.py –clocklist “./CCSLBenchmark/Spec3/clocklist” –spec “./CCSLBenchmark/Spec3/Spec3\_2.xml” --round 2000 --trace\_length* ***50//change here*** *–trace\_number 5 –trace\_directory “./CCSLBenchmark/Spec3/trace” –test “./CCSLBenchmark/Spec3/golden\_spec3\_2” –test\_times 500*

* **Spec4(5,1,1)**

*python main.py --clocklist "./CCSLBenchmark/Spec4/clocklist" --spec "./CCSLBenchmark/Spec4/Spec4\_2.xml" --round 2000 --trace\_length* ***50//change here*** *--trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec4/trace" --test "./CCSLBenchmark/Spec4/golden\_spec4\_2" --test\_times 500*

* **Spec5(1,1,5)**

*python main.py --clocklist "./CCSLBenchmark/Spec5/clocklist" --spec "./CCSLBenchmark/Spec5/Spec5\_1.xml" --round 2000 --trace\_length* ***50//change here*** *--trace\_number 5 --trace\_directory "./CCSLBenchmark/Spec5/trace" --test "./CCSLBenchmark/Spec5/golden\_spec5\_1" --test\_times 500*

* + 1. **Test Commands of Figure 6**

**We can complete the experiments in Figures 6 by changing the values of parameters “***--trace\_number***”.**

* **Spec1(0,1,2)**

*python main.py --clocklist "./CCSLBenchmark/Spec1/clocklist" --spec "./CCSLBenchmark/Spec1/spec1\_2.xml" --round 2000 --trace\_length 50 --trace\_number* ***5//change here*** *--trace\_directory "./CCSLBenchmark/Spec1/trace" --test "./CCSLBenchmark/Spec1/golden\_spec1\_2" --test\_times 500*

* **Spec2(4,2,4)**

*python main.py --clocklist "./CCSLBenchmark/Spec2/clocklist" --spec "./CCSLBenchmark/Spec2/Spec2\_2.xml" --round 2000 --trace\_length 50 --trace\_number* ***5//change here*** *--trace\_directory "./CCSLBenchmark/Spec2/trace" --test "./CCSLBenchmark/Spec2/golden\_spec2\_2" --test\_times 500*

* **Spec3(5,1,10)**

*python main.py –clocklist “./CCSLBenchmark/Spec3/clocklist” –spec “./CCSLBenchmark/Spec3/Spec3\_2.xml” --round 2000 --trace\_length 50 --trace\_number* ***5//change here*** *–trace\_directory “./CCSLBenchmark/Spec3/trace” –test “./CCSLBenchmark/Spec3/golden\_spec3\_2” –test\_times 500*

* **Spec4(5,1,1)**

*python main.py --clocklist "./CCSLBenchmark/Spec4/clocklist" --spec "./CCSLBenchmark/Spec4/Spec4\_2.xml" --round 2000 --trace\_length 50 --trace\_number* ***5//change here*** *--trace\_directory "./CCSLBenchmark/Spec4/trace" --test "./CCSLBenchmark/Spec4/golden\_spec4\_2" --test\_times 500*

* **Spec5(1,1,5)**

*python main.py --clocklist "./CCSLBenchmark/Spec5/clocklist" --spec "./CCSLBenchmark/Spec5/Spec5\_1.xml" --round 2000 --trace\_length 50 --trace\_number* ***5//change here*** *--trace\_directory "./CCSLBenchmark/Spec5/trace" --test "./CCSLBenchmark/Spec5/golden\_spec5\_1" --test\_times 500*