

Combinatorial Coverage Measurement Command Line Tool

Classic Mode

Classic mode is the command line version of CCM. It provides all the same functionality as the GUI version with some extra functionality such as reading ACTS input files to define the input domain. By default, classic mode is enabled.

Command Line Arguments:

To see a list of all the possible command line arguments run:

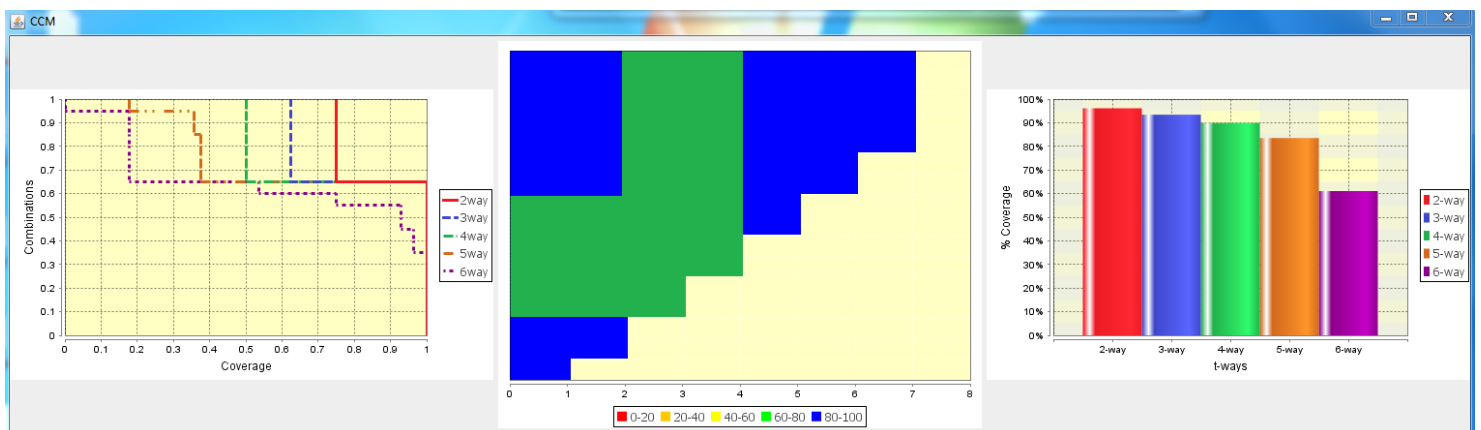
```
java -jar ccmcl.jar --help
```

To test the current version of the CCM command line tool download the ccmcl.jar file from GitHub. **EXAMPLE COMMAND LINE ARGUMENTS:**

```
C:\Windows\system32\cmd.exe

C:\Users\zbr\Desktop>java -jar ccmcl.jar -I input_test.csv -A ACTSfile.txt -P -T
2,3,4,5,6 -S -H -B
```

The output will be a graph like below depending on what charts you wish to display. Push “Ctrl+C” to stop the program.

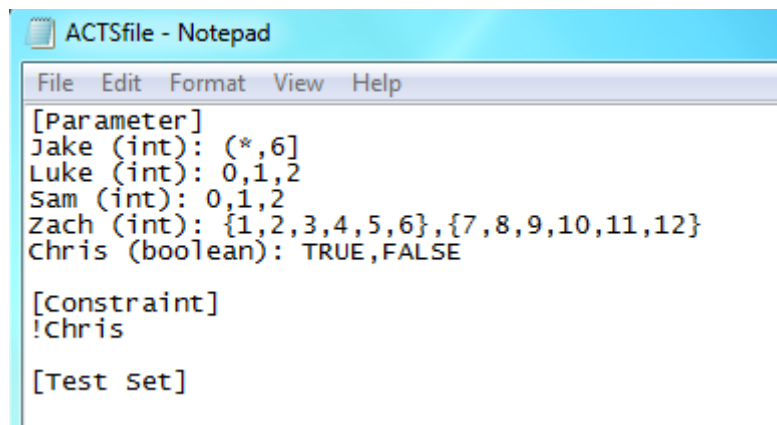


File types supported by CCM Command Line:

- .xml (define parameters, constraints, and test cases)
- .txt (define parameters, constraints, and test cases)
- .csv (defines test cases)

Range Classes

You can define a range class using interval notation. See parameter *Jake* in the ACTS configuration **.txt** file below:



```
ACTSfile - Notepad
File Edit Format View Help
[Parameter]
Jake (int): (*,6]
Luke (int): 0,1,2
Sam (int): 0,1,2
Zach (int): {1,2,3,4,5,6},{7,8,9,10,11,12}
Chris (boolean): TRUE,FALSE

[Constraint]
!Chris

[Test Set]
```

With interval notation, a boundary value at “6” was established. “*” values are used to represent negative infinity or positive infinity depending on the side of the interval. The notation used here represents that all values less than or equal to 6 go in group “0”, and all values greater than 6 go in group “1”. This is an important distinction to make for when we start using constraints.

Another example of range values can be seen below:

Jake (int): [1,8],[8,10],[13,*)

The interval notation above creates 4 boundary values effectively creating 5 different groups. To break it down:

- Group 1 – all numbers less than 1
- Group 2 – all numbers greater than or equal to 1, and less than 8
- Group 3 – all numbers greater than or equal to 8, and less than 10
- Group 4 – all numbers greater than or equal to 10, and less than 13
- Group 5 – all numbers greater than or equal to 13

For correct measurement, it is important to properly define the boundary values / range classes.

To define a range class in a .xml file, use **type = “3”**.

Group Classes

From the illustration on the previous page you can see that parameter *Zach* is a group class where values between { } are in a group together. {1,2,3,4,5,6} are said to be group “1”, and {7,8,9,10,11,12} are said to be group “2”.

To define a group in a .xml file use **type = “4”**.

Examples of Range and Group Values in .XML files

```
-----
<Parameters>
  <Parameter id="1" name="Jake" type="4">
    <values>
      <value>{0,1,2,3},{4,5,6}</value>
    </values>
    <basechoices>
      <basechoice />
    </basechoices>
  </Parameter>
  <Parameter id="2" name="Luke" type="0">
    <values>
      <value>0</value>
      <value>1</value>
      <value>2</value>
    </values>
    <basechoices>
      <basechoice />
    </basechoices>
  </Parameter>
  <Parameter id="3" name="Sam" type="0">
    <values>
      <value>0</value>
      <value>1</value>
      <value>2</value>
    </values>
    <basechoices>
      <basechoice />
    </basechoices>
  </Parameter>
  <Parameter id="4" name="Zach" type="3">
    <values>
      <value>(*,6]</value>
    </values>
    <basechoices>
      <basechoice />
    </basechoices>
  </Parameter>
```