

List of Known Toolbox Problems

17 November 2009

These are all the known bugs and missing features of the TLA⁺ Toolbox reported as of 17 November 2009.

1. New Feature

Add way to tell if the spec on which a model was last run is the same as the current spec.

5. New Feature (Locking Models)

TLC might be running for weeks, and the user in the meantime changed the spec. When TLC stops running, the model gets revalidated and stuff that was in the model configuration gets lost. Here is a possible solution: Add the notion of a model being **LOCKED**, meaning that it cannot be changed in any way, meaning that it is *read only* and it doesn't change to conform to the current spec. In particular, it conforms to the copy of the spec that's saved with the model. Running the spec locks the model. There will also be a menu item that the user can select to lock the spec himself. (This will allow the user to protect himself from accidentally modifying or re-running a spec by mistake—perhaps losing a trace that took weeks for TLC to produce.) We must decide what should happen to the lock state when a TLC run completes. Should the user have to explicitly unlock it? I suggest that it be unlocked when the run completes if and only if that run did not take too long—where “too long” is a user-settable preference.

9. Minor Missing Features (TLC Model)

The user cannot now declare the same model value in two different constant instantiations or in a constant instantiation and a definition override. Moreover, he cannot use a model value declared on the advanced page as a model value in a constant instantiation. These restrictions should be eliminated.

12. Minor Bug (Parsing Error View)

If the spec is reparsed after a parsing error, the Parsing Error view is displayed, and no errors are found, then the Parsing Error view continues to contain the previous error. The view has to be closed to make that error disappear.

13. Bug (PlusCal Translator Errors)

Suppose the user starts to write a PlusCal algorithm, translates it and gets a PlusCal error, and then decides to delete the PlusCal code and just write the spec in TLA^+ . There's no way for him to get rid of the PlusCal error message in the Parse Errors window. In fact, even if I run the translator on another module, the PlusCal error remains in the Parse Error window—whether or not the other translation produces an error. (The only way to get rid of it is to go back to the original module, correct the PlusCal error, and re-run the translator.) The best thing to do is probably to create a separate PlusCal Error Window. Running the translator should first clear this window and remove all PlusCal error markers in the module on which the translator is run. (I believe that if running the PlusCal translator creates a PlusCal error, then the file is not re-parsed, so only the PlusCal error window would need to be popped up.)

15. Possible Change (Handling Unparsed Spec)

We need to re-think what the user should be able to do to the model when the spec is unparsed; and see that anything he shouldn't be able to do is disabled and that anything he should be able to do works.

17. Minor Bug (Definition Override)

The Definitions Override menu allows the user to override the same definition multiple times.

18. Significant Bug (parsing large specs)

For a really big spec (the one we tried had dozens of files and over 7000 lines of TLA^+), after the spec is first parsed the parse spec command gets disabled. However, parse module file still works.

19. Minor Missing Feature (Exit)

Closing the toolbox simply kills any running TLC. Instead, it should warn the user that this will kill TLC and give him the option of canceling.

20. Missing Feature (PlusCal)

PlusCal translation needs to be examined. Here are some things that need to be fixed:

- There should be a more user-friendly way to specify translation options. This involves: (a) Providing check-boxes or radio buttons for the common options. (b) Finding a more convenient way to add them than the current one (right-clicking in the Toolbox Explorer) and selecting properties.
- Translation options are now attached to the spec; they should be attached to an individual module. (It's possible for a single spec to have PlusCal algorithms in two different modules.)

22. Bug (rename spec)

The rename spec command doesn't do the right thing if a spec by that name already exists.

27. Minor Bug (Parse Spec/Module Commands)

If I select Parse spec from the file menu when the file is dirty, a popup asks me if I want to save the resource. If I say yes, it doesn't save the file and parses the spec that's on disk. If I select Parse Module from the file menu when the module is dirty, it doesn't ask me if I want to save it. I expect it should.

28. Minor Bug (Reset Window Layout)

If I select `window/reset window layout` when the Welcome View should be shown, it produces a blank window instead of the Welcome View.

29. Feature (Module Editor Keybindings)

Here are some thoughts on enhancements to the Module Editor.

- Add a preference to choose an Emacs-like keybinding. This requires adding new commands that appear in the General/Editors/Keys, and options that choose from among sets of bindings.
- Add editing commands (and invent bindings) for comments.

Note that *toggle comments* is bound to Control-/, but that command doesn't appear in the Preferences menu's keybinding options.

32. Minor Bug (What to parse when)

The toolbox doesn't seem to be doing the right thing when deciding what to parse when the user requests that a module be parsed. This needs to be re-examined and specified precisely, and then we need to make sure that the implementation does what it's supposed to.

36. Feature (Action Fairness)

Enhance the Init/Next method of writing the spec to add lists of weakly and strongly fair actions.

39. Minor Feature (Constant Assignment)

When the user has assigned a set of model values to a constant and specified symmetry, the default on the next page of the wizard should be to make the values untyped.

40. Bug (Help)

The help menu is not getting the focus if it doesn't already have the focus and it either is in or is being opened into a folder that has the focus.

43. Minor Bug (Model Checking Results page display)

The layout/spacing of the statistics / progress output / user output fields sucks. It might be better to make a two-column view like in the other two pages, splitting the progress output into two separate fields, one in each column.

46. Bug (Model reports bogus "TLC still running")

Sometimes when restarting the toolbox after it was closed with a model

open, the model says that TLC is still running. The “stop TLC” button in the upper right is red, but clicking it does nothing. The last time this happened, the Spec Explorer said that the model has crashed, and clicking on repair solved the problem.

51. Very Minor Bug (Re-creating a spec)

When creating a new spec, the Import Existing spec option is always checked and disabled. It should be unchecked and disabled if the `.toolbox` directory does not already exist. However, if the `.toolbox` directory does exist, its present behavior of checking and enabling the box is the correct one.

53. Minor Bug (Eclipse resource saving?)

If I have automatic parse turned off, modify a file, and then try to parse it, the Toolbox pops up a dialog saying I have a modified resource and asking if I want to save it. When I say yes, it closes the dialog and goes ahead and parses the spec without saving the file.

54. Minor Bug (Open Module)

When the user clicks on File/Open Module/Choose TLA⁺ Module, the file browser that pops up should show the specification’s directory.

57. Bug (Pluscal Translator preferences)

The PlusCal translator preferences seem to be no-ops. The “Re-translate on module save” option should be made to work. This should cause the PlusCal translator to be called whenever a module is saved that contains the string `--algorithm`.

The “Popup problem window on translator errors” is currently redundant, since translator errors are shown in the Parsing Errors view, which has its own preference for controlling that.

63. Missing Feature (TLC Progress)

The Model Checking Results page should list in the General section the information about probability of fingerprint collision that TLC prints after a run that finishes computing the reachable states. (I think it prints it out in the event of a liveness error.) It should say approximately something like: Probability of fingerprint collision: theoretical: 4.3E-15 pragmatic: 8.7E-15

64. TLC Bug (Liveness checking in Simulation mode)

TLC’s liveness checking in simulation mode is seriously buggy. This is unlikely to be fixed soon, so we’ll have to tell users not to check liveness properties in simulation mode.

66. Minor Missing Feature (Results page/Simulation mode)

When TLC finds an error in simulation mode, it prints out a “seed” and “aril” that can be used to re-create the error. It would be nice if the Toolbox reported those values in the General section of the results page.

This is not very important because the Trace Explorer should make it unnecessary for the user to want to use them.

67. Missing Feature (saved spec files)

It should be possible for the user to open in a read-only editor window the copies of the module files saved when a model is run. The menu raised by right-clicking on the model in the Spec Explorer should have an “open saved module” option. Clicking on it should raise a file browser on the model’s directory that shows all the tla files there. (It wouldn’t hurt to allow the user to browse the MC.tla and even the MC.cfg file as well.)

68. Minor Missing Feature (MC file conflict)

We should change the model validator to check if the spec contains a module named MC and to pop up an error window if it does saying that the spec can’t be model checked.

71. Bug/Missing Feature (MC file parsing error bubbles.)

Putting the mouse cursor over a model error bubble produced by a parsing error in the MC file produces a message that mentions the location in the MC file. It needs to be translated to a location in the model.

76. Minor Bug (model editor Error blobs)

The Toolbox can detect an error in a model editor entry in two ways: by performing its own checks on it, and by validating the model (which involves parsing the MC file). The toolbox should remove the error marker for the first kind of error as soon as the user corrects it, but instead those markers are removed only when the model is validated.