

NAME

TranSyT – A tool for the verification of timed and untimed asynchronous concurrent Transition Systems

SYNOPSIS

TranSyT [-h][-f <filename>][-x <dbxlevel>]

A command-shell like tool to symbolically manipulate and verify hierarchical and coordinated Transition Systems based on the **TSIF** format.

Options:

- h Help information.
- f Executes the set of commands in the file <filename>.
- x Sets the debug flag level to <dbxlevel>.

DESCRIPTION

TranSyT is command shell that provides various functions to analyze, manipulate and verify hierarchical and coordinated Transition Systems described using the **TSIF** format. The tool is based on the efficient symbolic manipulation of large Transition Systems by means of Binary Decision Diagrams (CUDD and PDTRAV packages).

TOOL STRUCTURE

TranSyT allows the user to store multiple Transition Systems in a library of systems. Whenever a new hierarchical Transition System should be included in **TranSyT**, all its subcomponents must be already included in the library of systems. Note that circular references between systems are not supported.

A selected Transition Systems is always denoted to be the *default TS*. Many operation in **TranSyT** are designed to be specifically applied on the default TS. In general the last Transition System incorporated in the library of systems is selected as default TS. The result of some specific operations, that generate new Transition Systems, may change the default TS. However, at any time the user can designate any desired Transition System to be the default TS.

TranSyT includes a large set of operations that can be classified into the following categories:

General input/output: read and write **TSIF** systems, as well as other formalisms.

Miscellaneous: operations to configure the tool.

Basic TS manipulation: create new TS's connecting existing ones.

TS untimed verification: reachability analysis, on-the-fly invariant verification, hybrid verification, fair CTL model checking.

TS timed verification: invariant verification based on the Relative-time paradigm.

Information display: visualize traces and reachable states based on the **graphviz** toolkit.

GENERAL INPUT/OUTPUT COMMANDS

They provide support for including new Transition Systems in the library of systems, and to write into screen or into a file existing Transition Systems. An internal interface with other tools allows to automatically translate other models (such as Petri nets or digital circuits) into Transition Systems, or translate Transition Systems into other models such as **BLIF** and **SMV**.

read_ts: Reads a new Transition System.

- write_ts:** Writes an existing Transition System in **TSIF**, **BLIF** or **SMV** format.
- write_std:** Writes an State Transition Diagram (STD), an explicit state-based representation of an existing Transition System.
- read_pn:** Reads a Petri net in **ASTG** or **PEP** format via the **HCSTG** tool.
- read_blif:** Reads a digital circuit in **BLIF** format via the **MASC** tool or internal PDTRAV utilities.

MISCELLANEOUS COMMANDS

Various commands to configure the way **TranSyT** and the underlying CUDD packages works.

- show_ts:** Provides information on the Transition Systems stored in the library of systems.
- default_ts:** Sets the default Transition System to a system in the library or informs about the current default TS.
- read_cfg:** Reads configuration information that drives the verification process.
- write_cfg:** Writes configuration information that drives the verification process.
- dd_reorder:** Reorder the variables in the BDD package for the current TS.
- assign:** Sets various internal parameters or objects in **TranSyT**.
- source:** Executes the commands contained in a file, then returns to the interactive shell.
- help:** Provides online information on the available commands.
- quit:** Exits from **TranSyT**.

BASIC TS MANIPULATION COMMANDS

They offer basic transformational operations on Transition Systems. In general these commands generate new Transition Systems, the result of the desired transformation.

- delete_ts:** Removes a Transition System from the library of systems. The consistency with the rest of the library is checked.
- mirror:** Creates a mirror image of a Transition System, i.e. input variables and labels will become outputs and vice versa.
- flatten:** Selectively eliminates the hierarchy from a Transition System. It may require the reachability set of the system.
- compose:** Composes two Transition Systems creating a closed system.
- minimize:** Checks the redundancy level in the Transition System. Redundant internal variables can be eliminated from the system without modifying its behavior. Requires the reachability set of the system.
- abstract:** Eliminates labels (and its associated events) from a Transition System, abstracting its behavior. Requires the reachability set of the system. [Not implemented yet]

TS UNTIMED VERIFICATION AND FAILURE MANIPULATION COMMANDS

They perform different symbolically reachability algorithms on Transition Systems providing information on its reachability set, failure states, regions, etc.

- uverif:** Untimed verification of asynchronous system versus its specification.
- traverse:** Symbolically restores the reachability set without taking into account delay information. It may also analyze failure conditions on the fly.
- simulate:** Hybrid simulation guided-traversal strategy oriented to generate as much of the state space as possible.
- bug_hunt:** Hybrid simulation guided-traversal strategy oriented to detect timed and untimed failures as fast as possible.
- add_fail:** Adds selected failure conditions to the a Transition System.
- active_fail:** Activate/deactivate a selected failure condition.
- check_fails:** Checks the fail conditions of the default system.
- print_fails:** Shows the fail information stored for the specified system.

TS TIMED VERIFICATION COMMANDS

They combine untimed symbolically reachability analysis with min/max delay analysis in order to a Transition System contains failure states. All the min/max delay analysis is performed on demand and is introduced in the system by means of relative-time constraints, i.e. fixing the relative execution order of certain events.

- tverif:** Invariant verification based on the relative-time paradigm.
- prune_dr:** Invariant verification based on the relative-time paradigm.
- read_trace:** Read a trace stored in a text file for further manipulation.
- read_es:** Read an event structure in a text file for further manipulation.

INFORMATION DISPLAY COMMANDS

Various commands to visualize different types of system information. In some cases requires interface with the *graphviz* and *ghostview* packages.

- print_fails:** Prints on screen the status of the failure conditions defined for a Transition System.
- print_path:** Reports a path from a given set of reachable states to a state from the set of initial states. In general it can be use to generate witnesses of a given condition in the Transition System.
- print_states:** Prints on screen the characteristic function of a set of reachable states that satisfy specific conditions defined by the user.
- print:** Prints on screen the characteristic function of various objects inherent to the Transition System. It is independent of the actual reachability set.
- draw_std:** Graphically displays the State Transition Diagram (STD) associated to the reachability set (or portion) of a Transition System. Selected subsets of states may be highlighted to ease the analysis.

SYMBOLIC OBJECTS

The extended EQN format is used to describe characteristic functions that are always related to the default TS (see the **TSIF** formalism). The basic logic constructions are:

id	Variable name.
NS(id)	Next-state variable corresponding to a current-state variable.
inst.id	Variable 'id' inside an instance 'inst'.
eq'	Post-fix logic NOT.
!eq	Pre-fix logic NOT.
eq1*eq2	Logic AND (also EQN concatenation is accepted).
eq1+eq2	Logic OR.
eq1=eq2	Equivalence or logic XNOR.
eq1<>eq2	Difference or logic XOR.
eq1->eq2	Logic implication.
(eq)	Any level of parenthesis.

Extensions of this format include most objects used to symbolically model Transition Systems:

RS	Reachability set.
ISTATE	Initial state set.
PARTIAL	Set of partially reached states.
EN(lab)	Enabling function of label 'lab'.
EN(ev,lab)	Enabling function of event 'ev' from label 'lab'.
FF(lab)	Firing function of label 'lab'.
FF(ev,lab)	Firing function of event 'ev' from label 'lab'.
TR(lab)	Transition relation of label 'lab'.
TR(ev,lab)	Transition relation of event 'ev' from label 'lab'.
BTR(lab)	Inverse TR of label 'lab'.
BTR(ev,lab)	Inverse TR of event 'ev' from label 'lab'.
FL(lab)	Fail states of label 'lab' (for all defined conditions).
FL(ev,lab)	Fail states of event 'ev' from label 'lab' (for all defined conditions).
IMG(eq)	Image computation from 'eq'.
IMG <n> (eq)	A selected number <n> of image computation steps from 'eq'.
IMG(lab)(eq)	Image computation of label 'lab' from 'eq'.
PIMG(eq)	Pre-image computation from 'eq'.
PIMG(lab)(eq)	Pre-image computation of label 'lab' from 'eq'.

DETAILED COMMAND DESCRIPTION

Detailed description of all commands offered by **TranSyT** organized by topic.

GENERAL INPUT/OUTPUT COMMANDS

read_ts [<options>] <file-name>

Reads a <file-name> containing Transition Systems in **TSIF** format. The systems are added into the system library for future instantiations. All instances in a system must have been previously loaded. The last read Transition System is set as default TS.

Options:

-dbxN Temporal setting of the verbose level to N.

write_ts [<options>] [<ts-name>] [> <file-name>]

Writes in **TSIF** format the selected <ts-name> from the system library into the standard output or a selected file <file-name>.

Options:

-dbxN Temporal setting of the verbose level to N.

-r Writes any other TS required as instance.

-smv Writes in **SMV** format.

-blif Writes in **BLIF** format.

write_std [<options>] <ts-name> [> <file-name>]

Writes the selected <ts-name> from the system library into the standard output or a selected file <file-name>. The output is formatted using the state-based STD format.

Options:

-dbxN Temporal setting of the verbose level to N.

read_pn [<options>] <file-name> [<ifile-name>]

Reads a file containing Petri Net (or STG). Translation from **ASTG** into **TSIF** format is carried out by the HCSTG tool. The additional information contained in the <ifile-name> is also feed into HCSTG. The systems are added into the system database for future instantiations. Instances must have been previously loaded. The last read Transition System is set as default TS.

Options:

-dbxN Temporal setting of the verbose level to N.

-pep Reads the PN in **PEP** compatible format. -pn Force to read a PN (signals are not included).

-s Force the read STG to be encoded as safe.

-nL The interface does not contain visible labels.

-nS The interface does not contain visible variables.

read_blif [<options>] <file-name> [<ifile-name>] [<library-name>]

Reads a <file-name> containing a digital circuit in **BLIF** format. Translation from **BLIF** into **TSIF** format is carried out by the MASC tool. The system is added into the system library for future instantiations. The last read Transition System is set as default TS.

Options:

-dbxN Temporal setting of the verbose level to N.

-delay Include delays for input events (environment).

-nenv Do not include events for the environment.

-envdN Set the input (environment) delay to N.

-ext Read extra information such as initial values and delays in <ifile-name>.

-lib Read the specified gate library in <library-name>.

MISCELLANEOUS COMMANDS

delete_ts [<options>] <ts-name>

Deletes the specified <ts-name> Transition System. When no system is specified, the default TS is deleted.

Options:

-r Deletes all instances used by the TS but not used by other systems.

show_ts [<options>] [<ts-name>]

If a <ts-name> is specified offers detailed information for that particular system. Otherwise, writes the list of all Transition Systems that are present in the system library.

Options:

-d Offers a more detailed information about each Transition System.

default_ts [<ts-name>]

Sets <ts-name> as the default TS. In case no <ts-name> is specified, it shows the name of the current default TS.

read_cfg [<options>] [<file-name>]

Reads a file containing configuration information.

write_cfg [<options>] [> <file-name>]

Writes the configuration information from the system into the standard output or a selected file <file-name>.

dd_reorder [<options>]

Reorder the variables in the BDD package for the current TS.

Options:

-dbxN Temporal setting of the verbose level to N.

-IN Specifies the level of effort to be spent in the reordering.

-inf Writes statistics of the BDD manager.

-w Write the current variable order.

-smv Writes variables in format compatible to **NuSMV**.

-vis Writes variables in format compatible to **VIS**.

assign <symbol> <equation>

Assigns the specified <equation> to a certain <symbol>, in the context of the default model. The symbol can be used in other equations after being defined. Symbols can be freely redefined at any time.

The symbol PARTIAL can be defined by this mechanism (see the EQN format).

Options:

-v Shows all symbols defined for the default model.

assign option <parameter> <value>

Assigns values to internal parameters of the **TranSyT** tool. Both <parameter> and <value> must belong to the list of the valid parameters and values. If no parameter and value are specified the current values are shown.

Valid Parameters and Values are:

TRAV_METHOD:	BFS, CHAINED, TOKEN, WTOK, DEVCLU, CLUSTER, MPI
CLOSURE_METHOD:	NONE, T1, T2
DYN_REORDER:	OFF, ON
DYN_METHOD:	SIFT, GROUP_SIFT
DYN_LIMIT:	integer value
FAIL_METHOD:	OFF, ON, KSTEP, END
FAIL_LIMIT:	integer value
RESTRICT:	OFF, ON

source [<file-name>]

Executes the **TranSyT** commands in <file-name>.

help [<command>]

Lists all available commands in **TranSyT**. If a particular <command> is selected, more detailed information is shown.

quit

Exists from **TranSyT**.

BASIC TS MANIPULATION COMMANDS

mirror [<options>] <system-name> [<ts-name>]

Creates a mirror image of the specified <ts-name> by substituting input variables and labels by outputs and vice versa. If no <ts-name> is specified it takes the default Transition System. The name of the new TS is specified by the <system-name> parameter. Only asynchronous systems can be mirrored.

Options:

-dbxN Temporal setting of the verbose level to N.
-d Do not set the resulting system as default TS.

flatten [<options>] <system-name> [<ts-name>]

Creates a flatten version of the specified <ts-name> TS. If no <ts-name> is specified it takes the default Transition System. The name of new TS is specified by the <system-name> parameter.

Options:

- dbxN Temporal setting of the verbose level to N.
- d Do not set the resulting system as default TS.
- m Minimize the system after flattening.
- {type} Flattening algorithms, available types are:
 - sim Single event per synchronized label.
 - prj Based on event projection.
 - er Based on strongly connected excitation regions (multiple events).

compose [<options>] <env-ts-name> <impl-ts-name> [<system-name>]

Creates the composition of the <env-ts-name> and <impl-ts-name> TS's. <env-ts-name> is considered to be the environment of the implementation <impl-ts-name>. The name of new TS is specified by the <system-name> parameter.

Options:

- dbxN Temporal setting of the verbose level to N.
- d Do not set the resulting system as default TS.
- c Compose assuming that <env-ts-name> is the specification of the implementation.

minimize [<options>] [<ts-name>]

Eliminates redundant variables if they are not visible in the interface of the specified system <ts-name>. If no <ts-name> is specified it takes the default Transition System.

Options:

- dbxN Temporal setting of the verbose level to N.
- chk Redundant variables are only reported.

TS UNTIMED VERIFICATION AND FAILURE MANIPULATION COMMANDS**uverif [<options>] <ts-name-spec> <ts-name-impl>**

Untimed verification of a system <ts name impl.> versus its specification <ts-name-spec>.

Options:

- dbxN Temporal setting of the verbose level to N.
- VnotConformance
Do not check conformance of the implementation with respect to the given specification.
- VnotPersistence
Do not check persistency in the implementation.
- Vdeadlock
Check also the presence of dead-locks in the closed system.

- Vclose Build the closed system and create fails only.
- Vnotdestroy
Do not destroy intermediate TSs after verification.

traverse [<options>]

Performs a symbolic traversal of the default TS. All failure conditions specified in the system are checked on the fly. Different default options define how the traversal is implemented (that can be redefined with the assign command). Some of them can be overwritten in-line.

General Options:

- dbxN Temporal setting of the verbose level to N.
- i The system is only instantiated, no traversal is performed.
- p Use the partial computed states as initial states.

Traversal Options:

-T{algorithm}

Specifies the traversal scheme to be used. Note that further options may be available depending of the selected traversal scheme. Available schemes are the following:

- bfs Baseline BFS traverse algorithm.
- chained Mixed BFS/DFS traverse algorithm.
- token Token traverse algorithm.
- wtok Weighed token traverse algorithm.
- devclu Dynamic event clustered traverse algorithm.
- cluster Static event clustered traverse algorithm.
- closure Event closure traverse algorithm.
- step{k} K-step BFS traverse algorithm, where k is the number of performed steps.

-ST{param}

Parametrization of the existing traversal algorithms. Available parameters are the following:

- STorder Use a greedy heuristic to order the transition firing in the BFS/DFS traversal.
- STm The Monolithic TR is used in the BFS/DFS traversal.
- ST{h}l The label-partitioned TR is used in the BFS/DFS traversal. The -SThl flag allows to explore the TS hierarchy to minimize the BDD peak size (no failure analysis is implemented).
- ST{h}e The event-partitioned TR is used in the BFS/DFS traversal. The -SThe flag allows to explore the TS hierarchy to minimize the BDD peak size (no failure analysis is implemented).
- STstates Fire event with maximum states in devclu traversal algorithm.
- STnodes Fire event with maximum nodes in devclu traversal algorithm.
- STnolocal Fire event with maximum states, but minimizing number of BDDs in devclu traversal algorithm.

Failure Options:

- nF No failure analysis is implemented during the traversal.
- aF Check fails every time a TR is applied in the BFS/DFS traversal algorithm.
- eF Check fails just after the traverse (in all traversal algorithms).
- kF{k} Check fails every k iterations in the BFS/DFS traversal algorithm.
- R Failure states are restricted in the BFS/DFS traversal algorithm.
- nR Failure states are not restricted in the BFS/DFS traversal algorithm.

BDD Reordering Options:

- re Force the reorder of the BDD variables.
- reSIFT Force the reorder of the BDD variables using SIFT algorithm
- reGROUPSIFT
Force the reorder of the BDD variables using GROUPSIFT algorithm.
- limit{N}
Minimum number BDD nodes to apply reordering.

simulate [<options>]

Generates states via simulation of the default TS.

Options:

- dbxN Temporal setting of the verbose level to N.
- SdpN Max depth of states generated is N (default 40)
- SiteN Max number of states generated is N (default 1000)
- SbrN Max number of branches generated is N (default 5)
- SchN Max number of choices in one branch is N (default 15)
- SexpSIM
Expand the simulation via guided traversal (direct simulation).
- SexpES Expand the simulation via guided traversal (event structures).
- Sl2 Run two loops in the expansion process.
- Sfull Always keep the full trace in the simulation (default).
- Snfull Break the trace at the choice points.
- Ssatdelay
Satisfy the delay values to generate the trace.
- SNSatdelay
Do not satisfy the delay values to generate the trace.
- Ssymm Check conflicts to avoid asymmetric conflicts.
- Sfconc Check conflicts to select concurrency atoms first.
- Sfconf Check conflicts to select conflicting atoms first.

bug_hunt [<options>]

Detects failure conditions via simulation of the default TS.

- Options:
- dbxN Temporal setting of the verbose level to N.
 - SdpN Max depth of states generated is N (default 40)
 - SiteN Max number of states generated is N (default 1000)
 - SbrN Max number of branches generated is N (default 5)
 - SchN Max number of choices in one branch is N (default 15)
 - SexpSIM Expand the simulation via guided traversal (direct simulation).
 - SexpES Expand the simulation via guided traversal (event structures).
 - Sl2 Run two loops in the expansion process.
 - Sfull Always keep the full trace in the simulation (default).
 - Sfull Break the trace at the choice points.
 - Ssatdelay Satisfy the delay values to generate the trace.
 - SNSatdelay Do not satisfy the delay values to generate the trace.
 - Ssymm Check conflicts to avoid asymmetric conflicts.
 - Sfconc Check conflicts to select concurrency atoms first.
 - Sfconf Check conflicts to select conflicting atoms first.
 - SfailN Max number of fail states to be detected. (default 5).
 - SckdepthN Check failures after each N steps. (default 20).
 - Sckarc Check failure arcs every time an atom fires. (default false).
 - SrbdN Rebuild the failure trace using a window of size N. (default 5).
 - SWriteTraceN Write the traces in the selected format.

add_fail [<options>] <fail-type> [<name1>][,<name2>][EQN <equation>]

Adds selected failure conditions to the default system. Fails can be added to three types of objects <fail-type>:

TFAIL: to a transition system specified by <name1>

LFAIL: to a label specified by <name1>

EFAIL: to the event <name2> of label <name1>

The equation of the fail condition is specified by <equation>.

- Options:
- dbxN Temporal setting of the verbose level to N.
 - p Adds a persistency fail condition.
 - i ??Adds properties to internal labels??.
 - m Adds a conformance fail condition.
 - d Adds a deadlock fail condition.

active_fail [**<options>**] **<fail-type>** [**<name1>**][**<name2>**][**<fail num>**]

Activate/deactivate a selected failure condition. If no failure condition is selected, the operation will be applied to all failure conditions in the system (flag -a must be used).

Fails can be activated in three types of objects **<fail-type>**:

TFAIL: to a transition system instance specified by **<name1>**, e.g. **TFAIL(2)** to activate the fail condition two.

TFAIL: to a transition system instance specified by **<name1>**, e.g. **TFAIL(system.gate,1)** to activate the fail condition one at instance **<system.gate>**.

LFAIL: to a label specified by **<name1>**, e.g. **LFAIL(system.gate.out,1)** to activate the fail condition one at label **<out>** inside instance **<system.gate>**.

EFAIL: to the event **<name2>** of label **<name1>** e.g. **EFAIL(rise,system.gate.out,0)** to activate the fail condition zero at event **<rise>** of label **<out>** inside instance **<system.gate>**.

Options:

- dbxN Temporal setting of the verbose level to N.
- f Deactivate the failure condition.
- a Apply to all failure conditions in the system. Used to activate/deactivate all conditions in a single command.

check_fails [**<options>**]

Checks the fail conditions of the default system after it has been traversed.

Options:

- dbxN Temporal setting of the verbose level to N.

print_fails [**<options>**] [**<ts name>**]

Shows the fail information stored for the specified **<ts name>**. If no **<ts name>** is specified it takes the default TS.

Options:

- dbxN Temporal setting of the verbose level to N.
- e Prints information only for the events.
- l Prints information only for the labels.
- t Prints information only for the TSs.
- a Prints information for all fail conditions (by default only those with failure states are shown).
- s Prints the failing states.

TS TIMED VERIFICATION COMMANDS**tverif** [**<options>**] **<ts-name-spec>** **<ts-name-impl>**

Timed verification of a system **<ts-name-impl>** versus its specification **<ts-name-spec>**.

Specific options:

- VnotConformance
Do not check conformance of the implementation with respect to the given specification.
- VnotPersistency
Do not check persistency in the implementation.
- Vdeadlock
Check the presence of dead-locks in the system.

tverif [<options>] <ts name>

Timed verification of a system <ts name>.

Specific options:

- VnittersN
Perform only N iterations. If N=0 (default) iterate normally.

Common options of both tverif commands:

- dbxN Temporal setting of the verbose level to N.
- Vnotdestroy
Do not destroy intermediate TSs after verification.
- Vnzdp Do not perform atom's zero-delay pruning before starting verification. The pruning consists in restricting the Firing Functions of all the other concurrently enabled atoms such that those with zero delay fire first. Setting this flag may increase considerably the number of iterations required to perform the verification.
- pwp Perform atom's pairwise pruning before starting verification, based on their delays.
- VfailTrace
Indicates the way the failure trace is searched: (1) to perform a partial traversal; (2) perform a partial traversal using chaining; (3) to perform a fast simulation (bughunt).
- VextendTraceN
Extend traces following fails. N may take the values: 0 for no extension, 1 for extensions following a single fail, or 2 for multiple fails extensions (default N=0).
- VtraverseFreqN
Force reachability analysis every N iterations (default N=1).
- VreorderFreqN
Force reorder of BBD variables every N iterations (default N=0, i.e. no variable reorder).
- VminimizeFreqN
Force minimization of redundant BBD variables every N iterations (default N=0). N must be multiple of that specified with -VtraverseFreq option.
- VwriteTraceN
Write the failure trace up to the initial state at each iteration. N indicates the output format: 0 for no output, 1 .dot, 2 for .txt, and 3 for .xml . The different options are cumulative so that several outputs can be produced. Default value is 0.
- VwriteSuffixN
Write the portion of the failure trace used at each iteration. N indicates the output format: 0 for no output and 1 for .dot . outputs can be produced. Default value is 0.
- VwriteESN
Write the complete ES for the full failure trace, at each iteration.
- VwriteEESN
Write the complete extended ES for the full failure trace, at each iteration (see also options on building EESs).

-VwriteTESN

Write the timed ES portion at each iteration. N indicates the output format: 0 for no output, 1 .dot and 2 for .txt . The different options are cumulative so that several outputs can be produced. Default value is 0 in both cases.

-VwriteSTD

Write the resulting STD after each iteration.

Options applicable when building timed Event Structures from a Trace:

-Acapf Capture all fail states in the trace when building the ES.

-Apreconc

Capture ES preconcurrence following the trace.

-AtaComplete

Builds an ES for the complete trace, then perform timing analysis, etc. This may result interesting in some cases, however the fail removal is very local and thus it is not suitable for hard verification processes.

-AfailGuided

Uses the failure transitions as a source for heuristics which try to reduce the size of the ESs and focus more locally around the particular failure being analyzed.

-AfilterTedges

Tries to filter (remove) those timing edges which do not actually remove the fail from all the failure states of the trace. This is intended to reduce the size of the resulting timed ESs such that all the atoms that appear in it, are related to the fail being analyzed.

Options applicable when building Event Structures:

-Elnp Use local nodal points when possible.

-Enotgnp

Do not use global nodal points information. If this option is selected, -Elnp is selected automatically, otherwise the entry condition of the enabling compatible composition would not work in some cases.

-EenablingsN

Sets the level of detail to take into account in order to put non-enabling information to the GRC created from an ES. N=1 means to use the minimum of information, i.e. a fast algorithm but with the need of more extra encoding variables. N=2 means to use information from the trace used to build the ES (if any), i.e. a more complex algorithm but with almost no extra encoding. Default value is N=2.

-EcheckConc

Indicates whether not to consider only timing arcs between concurrently fireable vertexes (if set), but just consider any possible timing arc (if not set). By default all possible timing arcs are considered.

prune_dr [<options>] [<ts name>]

Pruning of untimed concurrency which is actually fake if delays are considered in <ts name>.

Options:

-dbxN Temporal setting of the verbose level to N.

-zdp Perform atom's zero-delay pruning. The pruning consists in restricting the Firing Functions of all the other concurrently enabled atoms such that those with zero delay fire first.

- npp Perform pairwise pruning without using ESs, only with the atoms around nodal points, if any.
- pwp Perform atom's pairwise pruning based on their delays. This also covers the previous case, but here timed event structures are used anyway, therefore increasing CPU cost and splitting.
- gwp Perform atom's group-wise pruning based on their delays.

read_trace [<options>] [<ts name>] [< <input file name>] [> <output file name>]

Reads a trace stored in a text file given by <input file name>. The trace is assumed to start from the initial state of the TS given by <ts name>. The file must conform to the plain text (.txt), or XML (.xml), formats described in the trace-format manual page. If an output file name is given, the trace is written to the file in the appropriate format according to the file name extension, including DOT (.dot) files.

Options:

- dbxN Temporal setting of the verbose level to N.
- RTlptc Check time-feasibility of the read trace, using LP.
- RTd Display the read trace.
- RTe Launch the trace editor with the read trace.
- RTdes Build and display an event structure for the whole trace.
- RTees Build an event structure for the whole trace and launch the ES editor.
- RTdtes Build and display an timed event structure for the whole trace.
- RTetes Build a timed event structure for the whole trace and launch the ES editor.
- RTbces Build a timed event structure for the smallest possible suffix of the trace, if possible, according to the compliance criteria defined in the TTA package, i.e. escape all fails, scape at least one fail, contradict the trace, etc. If the ES could be built it is displayed.
- RTextendN
Extends the read trace following the fail (N=1) or multiple fails (N=2).

read_es [<options>] [<ts name>] [< <input file name>] [< <input file name>] [> <output file name>]

Reads an Event Structure stored in a text file given by the first <input file name>. The ES is assumed to conform to the TS given by <ts name>. In order to provide more information for the generation of the GRC and the later composition, the trace from which the ES was built can be supplied. Such trace is given by the second <input file name> . The files must conform to the plain text (.txt), or XML (.xml), formats described in the es-format manual page. If an output file name is given, the ES is written to the file in the appropriate format according to the file name extension, including DOT (.dot) files.

Options:

- dbxN Temporal setting of the verbose level to N.
- RElptcN
Check time-feasibility of the read ES using LP. If N=1 disablings are not assumed to occur. If N=2 disablings are assumed to occur, which can make the LP problem infeasible. If the LP problem is feasible prints minimum and maximum firing times.

- RElptedgesN
Do timing analysis using LP and add the resulting timing edges to the ES. If N=1 disablings are not assumed to occur. If N=2 disablings are assumed to occur, which can make the LP problem infeasible.
- REtedges
Do timing analysis and add the resulting timing edges to the ES.
- REd Display the read ES.
- REe Launch the ES editor with the read ES.
- REdGRC
Build and display the corresponding GRC of the ES.
- REeGRC
Build the corresponding GRC of the ES and launch the GRC editor.
- REcompose
Does reachability analysis on the ES, encodes the resulting GRC,

INFORMATION DISPLAY COMMANDS

print_path [<options>] [**from** <equation>]

Finds an execution sequence from a given set of states to the initial state of the default TS. By default a state from the RS is taken. The state can be restricted by specifying an equation that forces selected values to the variables in the system.

Options:

- dbxN Temporal setting of the verbose level to N.
- d Display the path using the **graphviz** interface.
- e Builds the path based on events.
- l Builds the path based on labels.
- s Shows the reached states along the path.
- n Normalize the path if a failure is present (only available with -d).

print_states [<options>] [**-proj** <instance>] [**with** <equation>]

Prints a set of reachable states. By default the states in the RS are shown. The set of states can be restricted by specifying an <equation> that forces selected values to the variables in the system.

Options:

- dbxN Temporal setting of the verbose level to N.
- all Shows the selected states with respect all variables.
- num Shows the exact number of the states in the set.
- proj Projects the selected states to the variables in a particular <instance>.

print [<options>] <equation>

Prints the specified <equation> using the extended EQN format.

Options:

-dbxN Temporal setting of the verbose level to N.

draw_std [<options>] <ts-name> [subset <equation0>] [with <equation1>] [with2 <equation2>]
[from <equation3>] [> <file-name>]

Writes the selected <ts-name> from the system database using a state-based (STD) graphical format. The output can be written either to the standard output or a selected file <file-name>. If no output file is selected or the [-d] flag is activated, the system is also shown on the screen. Only the subset of states selected in <equation0> is displayed. Two subsets of states can be selected, <equation1> and <equation2>. All selected states and arcs between them are highlighted. A path to the initial state from a subset of states can be also constructed. The path starts from one of the selected states <equation3>, and all the arcs up to the initial state are highlighted.

Options:

-dbxN Temporal setting of the verbose level to N.

-p Use the PARTIAL set of states as reference

-T<type>

Selects the output file type (ps, gif or dot).

-b Binary: draw binary information associated to all variables.

-f Draw fail states in red color.

-bd Draw the border.

-sg State Graph: draw binary information associated to variables. with an equivalent label.

-h Draws a header including the system name, labels and displayed variables.

-d Forces the STD to be displayed on screen.

FILES

SEE ALSO

The **TranSyT** WEB page at: <http://research.ac.upc.es/VLSI/transyt/transyt.html>
tsif(1), blif(1), astg(1), hcstg(1), masc(1).

BUGS

Guess how many of them are hidden in the code. Bug reports will be welcomed (transyt@ac.upc.es), as well as requests and suggestions for improvement.

AUTHOR

Enric Pastor, Marco A. Pena and Marc Sole
Department of Computer Architecture
Universitat Politecnica de Catalunya
08034 Barcelona, Spain
e-mail: enric@ac.upc.es