Model Checking Contest 2015

Tool Submission Manual

http://mcc.lip6.fr

Contents

1 Introduction

This document presents the tool submission procedure of the Model Checking Contest @ Petri Nets 2015. Prior to any submission, please check that you meet the conditions of the Model Checking Contest @ Petri Nets 2015. These rules are available at `http://mcc.lip6.fr/rules.php`.

Please contact `Fabrice.Kordon@lip6.fr` if you have any question of if you find any inconsistency or problem in this document or in the procedure.

IMPORTANT: This year, tools will be processed on virtual machines having 4 cores + 4 GByte / core allocated. Planned confinement is 60 minutes per run (one examination on one instance of a model).

About the Execution Environment.

To improve the tool integration procedure, we developed a simple and separate benchmark environment to:

- enable the reproduction of the experience by anybody since, if you agree, submitted VM will be made publicly available,
- ease the work of tool developers when building their tool submission.

BenchKit[1], this benchmark environment, will be used for tool evaluation during the Model Checking Contest. Introduced for 2013 edition, it is being enhanced for the 2014 and 2015 editions. The tool submission kit embed simplified scripts from ***BenchKit***. If **KVM/Qemu**[2] or **VirtualBox**[3] is installed, then, you may operate and thus test the VM of your tool(s) in similar conditions that the ones of the MCC.

So, your tool submission is a disk image to be executed in a virtual machine. The disk image preferred format is `.vmdk` that is compatible at least with **KVM/Qemu** and **VirtualBox**.

2 Description of the 2015 Tool Submission Kit

This section presents the structure of the Tool Submission Kit and ends with a procedure to let you tool integrate your tool for a proper invocation during the evaluation phase of the MCC'2015.

2.1 Overview

Tools are operated in Virtual Machines (VM). Thus, the tool submission must be a disk-image containing your tool. By default, Linux is planned, and thus, the disk image we provide operates a Linux machine, if you need another distribution or another operating system, please have a look on section 4, page 7.

The tool submission kit is composed of the following elements:

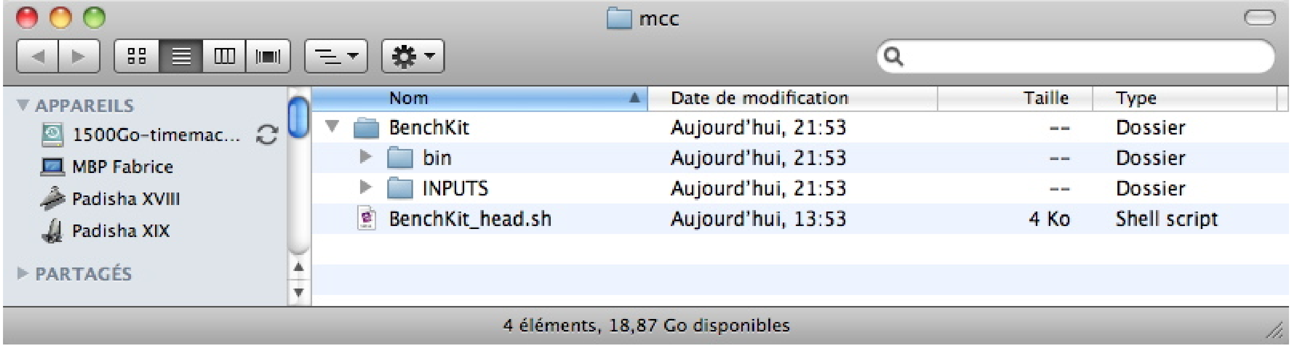
- A disk image preinstalled with models, formulas, and a dummy tool allowing tool developers to see how the system works. This dummy tool only supports the State Space examination for the first instance of each model and returns a result validated by tools in the previous years.
- a few scripts extracted from ***BenchKit*** to be operated with **KVM/Qemu** on a Linux machine or **VirtualBox** on a linux machine or a MacOS machine. Section 3, page 6, shows how to use this environment to test the behavior of the VM automatically in the conditions of the MCC'2015.
- the private `ssh` key (file `bk-private_key`) associated with the two accounts installed in the virtual machine (`mcc` and `root` are configured to log in with this key[4] – an empty passphrase is associated to this key). To connect with a password only, the password is: `mcc,2015`. Never remove this key because it will be used to operate your tool during the evaluation phase.

[1] ***BenchKit*** (`http://BenchKit.CosyVerif.org`) is developed within the context of the ***CosyVerif*** project (`http://CosyVerif.org`), supported by the **MeFoSyLoMa** group (`http://www.mefosyloma.fr`).

[2] `http://wiki.qemu.org`.

[3] `https://www.virtualbox.org`.

[4] To do so, start your connection as follows: `ssh -i bk-private_key ...`



Nom	Date de modification	Taille	Type
BenchKit	Aujourd'hui, 21:53	--	Dossier
bin	Aujourd'hui, 21:53	--	Dossier
INPUTS	Aujourd'hui, 21:53	--	Dossier
BenchKit_head.sh	Aujourd'hui, 13:53	4 Ko	Shell script

4 éléments, 18,87 Go disponibles

Figure 1: Structure of `mcc` user home directory in the disk image.

BenchKit and the provided disk image are ready to be used together, provided that you adapt the description of the machines to be used.

2.2 Content of the Disk Image

The disk image is a Linux system (Debian 7 – Wheezy stable – in 64 bit mode) including two accounts: `root` and `mcc`. The `mcc` home directory is structured as presented in figure 1. You find a unique directory, `BenchKit`, containing:

- `bin`, a directory where you should put all the executable and libraries of your tool. A dummy tool is provided as an example. It only supports state space generation for the first instances of each known model,

  You are free to install whatever you want in this directory that must contain all the libraries, executable files and data required to operate your tool.

- `INPUTS`, a directory that contains all the instances of models (there is one instance of model per value of the scaling parameters, only one when the model has no scaling parameter) provided to you. This directory contains one directory per input to be evaluated. Each directory contains a fixed number of files (PNML, properties, etc) that are detailed in section 2.5.

  Benchmark models are provided in a compressed archive to reduce the size of the disk image (one per model and per instance). Each archive contains all the required data for a given test (PNML files, formulas, etc.) When evaluating your tool for a given model instance and a given examination, **BenchKit** will uncompress the corresponding archive and execute your tool in that directory. See section 2.5, page 4 for more details on the content of each compressed directory.

- `BenchKit_head.sh`, a script executed remotely in the virtual machine; it is dedicated to the invocation of your tool.

  You must adapt this script that will be used by **BenchKit** to invoke appropriately your tool since the evaluation environment will only know it (and not command-line needed to run your tool). Two environment variables (see section 2.5) help you to determine which examination is being processed and, if several tools are hosted in the same VM, the tool to be invoked.

2.3 Overview of the Execution Procedure

The MCC'2015 execution procedure relies on **BenchKit**. Execution of your tool is driven by the script `BenchKit_head.sh` that is executed remotely on the virtual machine.

So, for each examination (state space generation, evaluation of properties, etc.), and each model instance, **BenchKit**:

- starts the virtual machine and uncompress the data required to operate the current examination.
- operates CPU and memory monitoring in the virtual machine,
- operates your tool for the examination on the model instance,
- stops the virtual machine[5] after retrieving the observed data and record them into a CSV file.

Please note that all tools are operated in the same conditions to avoid any deviation in the measurement of CPU and memory.

2.4 Connecting and Upgrading the Virtual Machine From the Disk Image

This section explains how you may operate the virtual machine from the disk image we provide. This is important to follows the integration directives provided in section 2.5.

To install extra software, you have access to the `root` account in this VM (see documentation). You must also install your stuff (binaries, extra model descriptions, etc.) in the `mcc` home directory (see section 2.5).

Starting the Virtual Machine. The following command starts manually a VM with your copy of the disk-image (let's call it `my-disk-image.vmdk`), please type:

```
$ qemu-kvm -vnc :42 -enable-kvm -smp 1 -cpu host -daemonize -k fr -m 2048 -drive file=my-disk-image.vmdk \
  -redir tcp:2222::22
```

You may want to omit the `-demonize` option to get the default screen where outputs of your VM are propagated. You may also connect using a VNC server (port 42).

This command supposes that you have **KVM/Qemu** but a similar operation can be done with **VirtualBox** (you must however set-up port redirection by means of the user interface before starting the VM).

If you want to connect without using ssh keys, please note that the **mcc** password is : `mcc,2015`.

Connecting to the Virtual Machine. Once the VM is running, you must type to log in:

```
$ ssh -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no -p 2222 -i bk-private_key mcc@localhost
```

where `bk-private_key` contains the private key associated to the public key installed for `mcc` (it is provided in the tool submission kit). The same couple of keys stand for `root`, thus allowing to install software if needed.

Please note that the options "-o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no" can be ignored. It is useful to avoid updating your `known_hosts` file and to have to change it if you manage several virtual machines with the same redirection port.

Copying files to/from the Virtual Machine. Once the VM is running, you must type to copy files:

```
$ scp -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no -P 2222 <files> mcc@localhost:<location>
```

where `<files>` represent your files and `<location>` the target destination of these files in the Virtual machine.

If you want to copy files without using ssh keys, please note that the **mcc** password is : `mcc,2015`.

[5] the `halt` command will be invoked from the root account, either if your tool terminates or if it timeouts

Value	Signification
StateSpace	we ask for state space generation only
ReachabilityDeadlock	Existence of deadlock states
ReachabilityFireabilitySimple	Firability of transitions
ReachabilityFireability	Boolean combinations of propositions checking firability of transitions
ReachabilityCardinality	Boolean combinations of propositions comparing the number of tokens in places
ReachabilityBounds	Boolean combinations of propositions comparing the bounds of places
ReachabilityComputeBounds	Computing the bounds of places
LTLFireabilitySimple	LTL properties with only one linear-time operator and no boolean operator
LTLFireability	Full LTL with atomic propositions checking firability of transitions
LTLCardinality	Full LTL with atomic propositions comparing the number of tokens in places
CTLFireabilitySimple	CTL properties with only one temporal operator and no boolean operator
CTLFireability	Full CTL with atomic propositions checking firability of transitions
CTLCardinality	Full CTL with atomic propositions comparing the number of tokens in places

Table 1: Possible values to refer examinations in the environment variable BK_EXAMINATION

2.5 Connecting Your Tool to the Execution Script

Since the connection of your tool to the execution system is based on the same principle as for the past editions, its adaptation for people who already participated in previous MCC should not be a problem. ***B*ench*K*it**, the execution environment sets-up environment variables to let you know what to do and launches your tool in a directory with all the appropriate data in.

Environment variables. There are two environment variables set-up for you by ***B*ench*K*it**:

- BK_EXAMINATION, that specifies the examination we expect your tool to perform. Possible values for this variable are presented in table 1, page 4. . Please have a look on the property language manual for more details on the corresponding classification
- BK_TOOL, that tells what tool is being processed by the system. This allows you to submit several tools (or variants of the same tool) hosted in the same image disk (then you have to clearly specify this when submitting your tools).

Content of the directory where your tool is being executed. As mentioned before BenchKit_head.sh is operated by ***B*ench*K*it** to run your tool in a directory that contains all you need to compute the current examination for one instance of a model:

- model.pnml, that is the initial PNML file that you may use for the model checking contest,
- iscolored, that contains either TRUE (if this is a colored model) or FALSE (if it is a P/T one),
- ⟨category⟩.xml and ⟨category⟩.txt where ⟨category⟩ is the value of the BK_EXAMINATION environment variable when it designates an examination where properties are required.

  Please note that the textual version of the formula is mainly to ease reading. As explained in the formula manual, the XML grammar of formula files only is provided (it is simpler to maintain and to use).
- equiv_col (for P/T nets) or equiv_pt (for Colored nets), a file containing FALSE (there is no equivalent colored or P/T net) or TRUE (there is an equivalent colored or P/T net),
- instance, a file containing the value of the current instance for the model.

When preparing your virtual machine, you may add dedicated files (e.g. adapted description of the model for your tool). **You are not allowed to cache pre-computed results.**

If you enrich the archive with your own description files, you must compress the directory and respect the naming convention and zip parameters (we use the `tar` command with `czf` arguments to compress, and `xzf` arguments to uncompress).

For "known" Models. you are allowed to enrich the content of this directory as long as you keep the conventions in the compressed archive replacing the original one. If PNML files are useless (they may be large), you are allowed to remove them as long as it does not hinder the correct execution of your tool. **Remind that precomputation of results is not allowed, only the information stored in model forms can be exploited.**

For "surprise" Models. "surprise" models are meant to evaluate tools in a default mode (*i.e.* no dedicated optimization. Thus, they are not present in the initial archive provided to you but they have the exact same structure and an empty file named `NewModel` will be located in the directory in that case to let your tool detect such situations.

For these models, only a PNML description will be provided and thus, to participate in this category, your tool will be required to import the PNML format.

2.6 Answering to *B*ench*K*it and the Dedicated MCC'2015 Post-Analysis Scripts

Your tool must answer in `stdout` and may provide alternative messages on `stderr` too. Both will be reported separately (this is useful for the debug phase).

However, there should be a dedicated line strictly respecting the format dedicated to a given examination. This line must start with dedicated keywords (see below). These keyword must not appear on the head of a line neither in `stdout` nor `stderr`.

The format your tool must respect when answering the examinations is presented below.

When an examination is not supported. The output on `stdout` must contain the following line:

```
DO_NOT_COMPETE
```

When the tool crashes (and you detect it). the output on `stdout` when you detect that your tool crashes must contain the following line:

```
CANNOT_COMPUTE
```

State Space generation. The output on `stdout` for this examination (in `benchKit_head.sh`, `BK_EXAMINATION="StateSpace"`) must contain the following lines:

`STATE_SPACE STATES` $\langle num_S \rangle$ `TECHNIQUES` $\langle technique_1 \rangle$ ... $\langle technique_n \rangle$

`STATE_SPACE TRANSITIONS` $\langle num_T \rangle$ `TECHNIQUES` $\langle technique_1 \rangle$ ... $\langle technique_n \rangle$

`STATE_SPACE MAX_TOKEN_PER_MARKING` $\langle num_M \rangle$ `TECHNIQUES` $\langle technique_1 \rangle$ ... $\langle technique_n \rangle$

`STATE_SPACE MAX_TOKEN_IN_PLACE` $\langle num_P \rangle$ `TECHNIQUES` $\langle technique_1 \rangle$ ... $\langle technique_n \rangle$

where $\langle num_S \rangle$ is the number of states found in the marking graph, $\langle num_T \rangle$ is the number of transitions firing in the marking graph, $\langle num_M \rangle$ is the maximum number of tokens per marking in the net, $\langle num_P \rangle$ is the maximum number of token that can be found in a place and where $\langle technique_i \rangle$ describes the verification technique(s) that has(ve) been used by your tool. Please pick one value in the set presented in table 2, page 6. Each technique must be separated by a space. You must specify several techniques if needed (no more than 6 please).

These lines can be provided in any order.

However, only the computation of $\langle number_S \rangle$ is mandatory. Tools may answer -1 for $\langle num_T \rangle$, $\langle num_M \rangle$, and $\langle num_P \rangle$ if they cannot compute the information (providing these values brings a bonus).

Any Examination Involving Properties. The output on `stdout` for this examination (in `benchKit_head.sh`, `BK_EXAMINATION`≠”StateSpace”) must contain the following line:

Value	Signification
ABSTRACTIONS	your tool exploits the use of abstractions (on the fly state elimination)
DECISION_DIAGRAMS	your tool uses any kind of decision diagrams
EXPLICIT	your tool does explicit model checking
NET_UNFOLDING	your tool uses McMillan unfolding
UNFOLDING_TO_PT	your tool transforms colored nets into their equivalent P/T
PARALLEL_PROCESSING	your tool uses multithreading (four cores will be allocated to the VM)
STRUCTURAL_REDUCTION	your tool uses structural reductions (Berthelot, Haddad, etc.)
SAT_SMT	your tool uses a constraint solver
STATE_COMPRESSION	your tool uses some compression technique (other than decision diagrams)
STUBBORN_SETS	your tool uses partial order technique
SYMMETRIES	your tool exploits symmetries of the system
TOPOLOGICAL	your tool uses structural informations on the Petri net itself (e.g. siphons, traps, S-invariants or T-invariants, etc.) to optimize model checking
USE_NUPN	the PNML model contains a NUPN-toolspecific section (see `http://mcc.lip6.fr/nupn.php`) and your tool takes advantage of it

Table 2: List of possible techniques identified to characterize your tool. If some technique you use is not referenced, please contact us.

`FORMULA` $\langle name \rangle$ $\langle res \rangle$ `TECHNIQUES` $\langle technique_1 \rangle$ ... $\langle technique_n \rangle$

where $\langle name \rangle$ is the formula identifier (provided in both the XML and textual format, see the formula manual for more details) and $\langle res \rangle$ the result of the formula:

- for the `ReachabilityComputeBounds` examination(see table 1), the result is an integer value,
- for other formulas, the result is a boolean and we expect `TRUE` or `FALSE`.

There must be one such line per formula in the file. A fine classification of formulas is proposed so that tools should only participate when they can handle the class of formulas that will be presented in his file (see the property language manual for more details about the subcategories of the contest).

For each formula your tool has a problem, please return :

`FORMULA` $\langle name \rangle$ `CANNOT_COMPUTE`

IMPORTANT: never answer `DO_NOT_COMPETE` for a given formula since this keyword must be only used to state that a tool does not participate in the whole subcategory.

Identifiers for Involved Techniques. Table 2 presents the list of identified techniques that could characterize your tool. If your tool uses a technique that is clearly not presented here, please add an appropriate keyword (one identifier, possibly containing the _ character) and provide us with a short explanation of this technique to update the table.

If your tool relies on another formalism than Petri net, you may provide the name of the formalism as a technique. Then, please put it in the first position.

3 Testing the Virtual Machine in the MCC'2015 Conditions

To launch an execution of your tool in the conditions of the MCC'2015, your must let the structure of the tool submission kit unchanged and be in the root of the uncompressed archive.

The main script to be used is `BenchKitStart.sh`. This script boots a VM with your disk image, then operate your tool for a given examination on a given model and then stops the VM and display the outputs of your tool.

To be operated, it requires either **KVM/Qemu** or **VirtualBox** to be installed on your machine. Then, you can check the behavior of your tool in the conditions of the MCC'2015. You can also check that outputs conform to the expectations of section 2.6.

It requires four mandatory parameters:

- the path of the disk image to be booted and executed by the VM containing your tool,
- the value to be affected to the `BK_EXAMINATION` environment variable defining what operation is to be executed on the VM (see table 1, page 4),
- the name of the tool to be invoked,
- the name of model to be processed with the examination (possible values are provided in table 3, Appendix A, page 8).

Thus, a typical invocation is:

`./BenchKitStart.sh` ⟨*vmname*⟩`.vmdk` ⟨*examination*⟩ ⟨*toolName*⟩ ⟨*modelName*⟩

A full execution example is provided in Appendix B, page 8.

4 Creating your Own Disk Image

When creating your disk image, please be sure it emulates a 64bits machine and has 5 GBytes free in the filesystem. It is also advised that you avoid this image to be more than 2 GBytes.

If you provide your own customized disk image, it must respect the following requirements:

- The logins *mcc* and *root* must be installed in the exact way they are in the disk image we distribute. In particular, the public ssh-key must be appropriately installed for the two logins and the machine must be reachable using `ssh`.

  If your disk image runs under Windows, please contact us (`Fabrice.Kordon@lip6.fr` and `Francis.Hulin-Hubard@lsv.ens-cachan.fr`)
- You must untar, in the home directory, the content of the archive provided here: `http://mcc.lip6.fr/archives/MCC-INPUTS.tgz`. It contains the structure of the input models to be installed in the `mcc` account. You must add the `bin` directory as well as your copy of the `BenchKit_head.sh` file.
- Install all packages required for your tool to run.

A Appendix – the names of "known" models

This appendix displays all the model/instances that are provided this year in the "known" model category. It allows you to check automatically the behavior of your tool submission in the conditions of the Model Checking Contest.

Name of the model/instances (known models)		
ARMCacheCoherence-PT-none	Angiogenesis-PT-01	Angiogenesis-PT-05
Angiogenesis-PT-10	Angiogenesis-PT-15	Angiogenesis-PT-20
Angiogenesis-PT-25	Angiogenesis-PT-50	CSRepetitions-COL-02
CSRepetitions-COL-03	CSRepetitions-COL-04	CSRepetitions-COL-05
CSRepetitions-COL-07	CSRepetitions-COL-10	CSRepetitions-PT-02
CSRepetitions-PT-03	CSRepetitions-PT-04	CSRepetitions-PT-05
CSRepetitions-PT-07	CSRepetitions-PT-10	CircadianClock-PT-000001
CircadianClock-PT-000010	CircadianClock-PT-000100	CircadianClock-PT-001000
CircadianClock-PT-010000	CircadianClock-PT-100000	CircularTrains-PT-012
CircularTrains-PT-024	CircularTrains-PT-048	CircularTrains-PT-096
CircularTrains-PT-192	CircularTrains-PT-384	CircularTrains-PT-768
DatabaseWithMutex-COL-02	DatabaseWithMutex-COL-04	DatabaseWithMutex-COL-10
DatabaseWithMutex-COL-20	DatabaseWithMutex-COL-40	DatabaseWithMutex-PT-02
DatabaseWithMutex-PT-04	DatabaseWithMutex-PT-10	DatabaseWithMutex-PT-20
DatabaseWithMutex-PT-40	Dekker-PT-010	Dekker-PT-015
Dekker-PT-020	Dekker-PT-050	Dekker-PT-100
Dekker-PT-200	Diffusion2D-PT-D05N010	Diffusion2D-PT-D05N050
Diffusion2D-PT-D05N100	Diffusion2D-PT-D05N150	Diffusion2D-PT-D05N200
Diffusion2D-PT-D05N250	Diffusion2D-PT-D05N300	Diffusion2D-PT-D05N350
Diffusion2D-PT-D10N010	Diffusion2D-PT-D10N050	Diffusion2D-PT-D10N100
Diffusion2D-PT-D10N150	Diffusion2D-PT-D10N200	Diffusion2D-PT-D20N010
Diffusion2D-PT-D20N050	Diffusion2D-PT-D20N100	Diffusion2D-PT-D20N150
Diffusion2D-PT-D30N010	Diffusion2D-PT-D30N050	Diffusion2D-PT-D30N100
Diffusion2D-PT-D30N150	Diffusion2D-PT-D40N010	Diffusion2D-PT-D40N050
Diffusion2D-PT-D40N100	Diffusion2D-PT-D40N150	Diffusion2D-PT-D50N010
Diffusion2D-PT-D50N050	Diffusion2D-PT-D50N100	Diffusion2D-PT-D50N150
DotAndBoxes-COL-2	DotAndBoxes-COL-3	DotAndBoxes-COL-4
DotAndBoxes-COL-5	DrinkVendingMachine-COL-02	DrinkVendingMachine-COL-10
DrinkVendingMachine-COL-16	DrinkVendingMachine-COL-24	DrinkVendingMachine-COL-48
DrinkVendingMachine-COL-76	DrinkVendingMachine-COL-98	DrinkVendingMachine-PT-02
ERK-PT-000001	ERK-PT-000010	ERK-PT-000100
ERK-PT-001000	ERK-PT-010000	ERK-PT-100000
Echo-PT-d02r09	Echo-PT-d02r11	Echo-PT-d02r15
Echo-PT-d02r19	Echo-PT-d03r03	Echo-PT-d03r05
Echo-PT-d03r07	Echo-PT-d04r03	Echo-PT-d05r03
EnergyBus-PT-none	Eratosthenes-PT-010	Eratosthenes-PT-020
Eratosthenes-PT-050	Eratosthenes-PT-100	Eratosthenes-PT-200
Eratosthenes-PT-500	FMS-PT-002	FMS-PT-005
FMS-PT-010	FMS-PT-020	FMS-PT-050
FMS-PT-100	FMS-PT-200	FMS-PT-500
GlobalResAllocation-COL-03	GlobalResAllocation-COL-05	GlobalResAllocation-COL-06
GlobalResAllocation-COL-07	GlobalResAllocation-COL-09	GlobalResAllocation-COL-10
GlobalResAllocation-COL-11	GlobalResAllocation-PT-03	HouseConstruction-PT-002
HouseConstruction-PT-005	HouseConstruction-PT-010	HouseConstruction-PT-020

Table 3: names of all the model/instances of "known" models

Name of the model/instances (known models)		
HouseConstruction-PT-050	HouseConstruction-PT-100	HouseConstruction-PT-200
HouseConstruction-PT-500	IBMB2S565S3960-PT-none	Kanban-PT-0005
Kanban-PT-0010	Kanban-PT-0020	Kanban-PT-0050
Kanban-PT-0100	Kanban-PT-0200	Kanban-PT-0500
Kanban-PT-1000	LamportFastMutEx-COL-2	LamportFastMutEx-COL-3
LamportFastMutEx-COL-4	LamportFastMutEx-COL-5	LamportFastMutEx-COL-6
LamportFastMutEx-COL-7	LamportFastMutEx-COL-8	LamportFastMutEx-PT-2
LamportFastMutEx-PT-3	LamportFastMutEx-PT-4	LamportFastMutEx-PT-5
LamportFastMutEx-PT-6	LamportFastMutEx-PT-7	LamportFastMutEx-PT-8
MAPK-PT-008	MAPK-PT-020	MAPK-PT-040
MAPK-PT-080	MAPK-PT-160	MAPK-PT-320
MultiwaySync-PT-none	NeoElection-COL-2	NeoElection-COL-3
NeoElection-COL-4	NeoElection-COL-5	NeoElection-COL-6
NeoElection-COL-7	NeoElection-COL-8	NeoElection-PT-2
NeoElection-PT-3	NeoElection-PT-4	NeoElection-PT-5
NeoElection-PT-6	NeoElection-PT-7	NeoElection-PT-8
ParamProductionCell-PT-0	ParamProductionCell-PT-1	ParamProductionCell-PT-2
ParamProductionCell-PT-3	ParamProductionCell-PT-4	ParamProductionCell-PT-5
PermAdmissibility-COL-01	PermAdmissibility-COL-02	PermAdmissibility-COL-05
PermAdmissibility-COL-10	PermAdmissibility-COL-20	PermAdmissibility-COL-50
PermAdmissibility-PT-01	PermAdmissibility-PT-02	PermAdmissibility-PT-05
PermAdmissibility-PT-10	PermAdmissibility-PT-20	PermAdmissibility-PT-50
Peterson-COL-2	Peterson-COL-3	Peterson-COL-4
Peterson-COL-5	Peterson-COL-6	Peterson-COL-7
Peterson-PT-2	Peterson-PT-3	Peterson-PT-4
Peterson-PT-5	Peterson-PT-6	Peterson-PT-7
Philosophers-COL-000005	Philosophers-COL-000010	Philosophers-COL-000020
Philosophers-COL-000050	Philosophers-COL-000100	Philosophers-COL-000200
Philosophers-COL-000500	Philosophers-COL-001000	Philosophers-COL-002000
Philosophers-COL-005000	Philosophers-COL-010000	Philosophers-COL-050000
Philosophers-COL-100000	Philosophers-PT-000005	Philosophers-PT-000010
Philosophers-PT-000020	Philosophers-PT-000050	Philosophers-PT-000100
Philosophers-PT-000200	Philosophers-PT-000500	Philosophers-PT-001000
Philosophers-PT-002000	Philosophers-PT-005000	Philosophers-PT-010000
PhilosophersDyn-COL-03	PhilosophersDyn-COL-10	PhilosophersDyn-COL-20
PhilosophersDyn-COL-50	PhilosophersDyn-COL-80	PhilosophersDyn-PT-03
PhilosophersDyn-PT-10	PhilosophersDyn-PT-20	Planning-PT-none
PolyORBLF-COL-S02J04T06	PolyORBLF-COL-S02J04T08	PolyORBLF-COL-S02J04T10
PolyORBLF-COL-S02J06T06	PolyORBLF-COL-S02J06T08	PolyORBLF-COL-S02J06T10
PolyORBLF-COL-S04J04T06	PolyORBLF-COL-S04J04T08	PolyORBLF-COL-S04J04T10
PolyORBLF-COL-S04J06T06	PolyORBLF-COL-S04J06T08	PolyORBLF-COL-S04J06T10
PolyORBLF-COL-S06J04T04	PolyORBLF-COL-S06J04T06	PolyORBLF-COL-S06J04T08
PolyORBLF-COL-S06J06T04	PolyORBLF-COL-S06J06T06	PolyORBLF-COL-S06J06T08
PolyORBLF-PT-S02J04T06	PolyORBLF-PT-S02J04T08	PolyORBLF-PT-S02J04T10
PolyORBLF-PT-S02J06T06	PolyORBLF-PT-S02J06T08	PolyORBLF-PT-S02J06T10
PolyORBLF-PT-S04J04T06	PolyORBLF-PT-S04J04T08	PolyORBLF-PT-S04J04T10
PolyORBLF-PT-S04J06T06	PolyORBLF-PT-S04J06T08	PolyORBLF-PT-S04J06T10
PolyORBLF-PT-S06J04T04	PolyORBLF-PT-S06J04T06	PolyORBLF-PT-S06J04T08
PolyORBLF-PT-S06J06T04	PolyORBLF-PT-S06J06T06	PolyORBLF-PT-S06J06T08

Table 3: names of all the model/instances of "known" models

Name of the model/instances (known models)		
PolyORBNT-COL-S05J20	PolyORBNT-COL-S05J30	PolyORBNT-COL-S05J40
PolyORBNT-COL-S05J60	PolyORBNT-COL-S05J80	PolyORBNT-COL-S10J20
PolyORBNT-COL-S10J30	PolyORBNT-COL-S10J40	PolyORBNT-COL-S10J60
PolyORBNT-COL-S10J80	PolyORBNT-PT-S05J20	PolyORBNT-PT-S05J30
PolyORBNT-PT-S05J40	PolyORBNT-PT-S05J60	PolyORBNT-PT-S05J80
PolyORBNT-PT-S10J20	PolyORBNT-PT-S10J30	PolyORBNT-PT-S10J40
PolyORBNT-PT-S10J60	PolyORBNT-PT-S10J80	ProductionCell-PT-none
QuasiCertifProtocol-COL-02	QuasiCertifProtocol-COL-06	QuasiCertifProtocol-COL-10
QuasiCertifProtocol-COL-18	QuasiCertifProtocol-COL-22	QuasiCertifProtocol-COL-28
QuasiCertifProtocol-COL-32	QuasiCertifProtocol-PT-02	QuasiCertifProtocol-PT-06
QuasiCertifProtocol-PT-10	QuasiCertifProtocol-PT-18	QuasiCertifProtocol-PT-22
QuasiCertifProtocol-PT-28	QuasiCertifProtocol-PT-32	Railroad-PT-005
Railroad-PT-010	Railroad-PT-020	Railroad-PT-050
Railroad-PT-100	ResAllocation-PT-R002C002	ResAllocation-PT-R003C002
ResAllocation-PT-R003C003	ResAllocation-PT-R003C005	ResAllocation-PT-R003C010
ResAllocation-PT-R003C015	ResAllocation-PT-R003C020	ResAllocation-PT-R003C050
ResAllocation-PT-R003C100	ResAllocation-PT-R005C002	ResAllocation-PT-R010C002
ResAllocation-PT-R015C002	ResAllocation-PT-R020C002	ResAllocation-PT-R050C002
ResAllocation-PT-R100C002	Ring-PT-none	RwMutex-PT-r0010w0010
RwMutex-PT-r0010w0020	RwMutex-PT-r0010w0050	RwMutex-PT-r0010w0100
RwMutex-PT-r0010w0500	RwMutex-PT-r0010w1000	RwMutex-PT-r0010w2000
RwMutex-PT-r0020w0010	RwMutex-PT-r0100w0010	RwMutex-PT-r0500w0010
RwMutex-PT-r1000w0010	RwMutex-PT-r2000w0010	SharedMemory-COL-000005
SharedMemory-COL-000010	SharedMemory-COL-000020	SharedMemory-COL-000050
SharedMemory-COL-000100	SharedMemory-COL-000200	SharedMemory-COL-000500
SharedMemory-COL-001000	SharedMemory-COL-002000	SharedMemory-COL-005000
SharedMemory-COL-010000	SharedMemory-COL-020000	SharedMemory-COL-050000
SharedMemory-COL-100000	SharedMemory-PT-000005	SharedMemory-PT-000010
SharedMemory-PT-000020	SharedMemory-PT-000050	SharedMemory-PT-000100
SimpleLoadBal-COL-02	SimpleLoadBal-COL-05	SimpleLoadBal-COL-10
SimpleLoadBal-COL-15	SimpleLoadBal-COL-20	SimpleLoadBal-PT-02
SimpleLoadBal-PT-05	SimpleLoadBal-PT-10	SimpleLoadBal-PT-15
SimpleLoadBal-PT-20	Solitaire-PT-EngCT7x7	Solitaire-PT-EngNC7x7
Solitaire-PT-FrnCT7x7	Solitaire-PT-FrnNC7x7	Solitaire-PT-SqrCT5x5
Solitaire-PT-SqrNC5x5	TokenRing-COL-005	TokenRing-COL-010
TokenRing-COL-015	TokenRing-COL-020	TokenRing-COL-030
TokenRing-COL-040	TokenRing-COL-050	TokenRing-COL-100
TokenRing-COL-200	TokenRing-COL-500	TokenRing-PT-005
TokenRing-PT-010	TokenRing-PT-015	TokenRing-PT-020
TokenRing-PT-030	TokenRing-PT-040	UtahNoC-PT-none
Vasy2003-PT-none		

Table 3: names of all the model/instances of "known" models

B Appendix – An invocation example

We provide below an example of execution with our dummy tool.

```
[fko TToolSubmissionKit]$ time ./BenchKitStart.sh mcc2015.vmdk StateSpace dummyTool PolyORBLF-COL-S02J04T06
   no memory confinment provided, assuming 1024 MBytes
   no VNC port specified, assuming 42
   no ssh redirection port specified, assuming 2222

execution on quadhexa-2.u-paris10.fr (runId=testing-run)
=====================================================================
runnning dummyTool on PolyORBLF-COL-S02J04T06 (StateSpace)
Warning: Permanently added '[localhost]:2222' (ECDSA) to the list of known hosts.
We got on stdout:
Probing ssh
Waiting ssh to respond
Ssh up and responding
=====================================================================
 Generated by BenchKit version MCC2015 (monitoring deactivated, Feb 10, 2015)
    Executing tool dummyTool:
    Test is PolyORBLF-COL-S02J04T06, examination is StateSpace
=====================================================================

--------------------
content from stdout:

START 1423653909
======================================================
== this is MyTool, a dummy example for the MCC'2015 ==
======================================================
Runing PolyORBLF (COL), instance S02J04T06

This tool just provides known information about state space of known
models for the first instance. This information comes from  the past
editions  of the  model checking contest  and is provided to let you
check that your tool provides appropriate results

STATE_SPACE STATES 104388 TECHNIQUES DUMMY_TECHNIQUE1 DUMMY_TECHNIQUE2
STATE_SPACE TRANSITIONS 193716 TECHNIQUES DUMMY_TECHNIQUE1
STATE_SPACE MAX_TOKEN_PER_MARKING -1 TECHNIQUES DUMMY_TECHNIQUE2
STATE_SPACE MAX_TOKEN_IN_PLACE -1 TECHNIQUES DUMMY_TECHNIQUE1 DUMMY_TECHNIQUE2
STOP 1423653909

--------------------
content from stderr:


--------------------
content from /tmp/BenchKit_head_log_file.1651:


real 0m23.504s
user 0m0.180s
sys  0m0.030s
```

The `--help` argument produces a small help as shown below

```
[fko  ./BenchKitStart.sh --help
usage: ./BenchKitStart.sh [-m <val>] [-vnc <val>] [-ssh <val>] <disk-image> <bk-examination> <tool-name> <input>
       -m: <val> Mbyte of memory confinment are assumed (default is 1024)
     -vnc: <val> is the VNC port for the launched VM (default is 42)
     -ssh: <val> is the SSH port for the launched VM (default is 2222)
     <disk-image>     : the path of the disk image to be booted and executed by the VM
     <bk-examination> : see BenchKit documentation, the variable defining what operation is to be executed on the VM
     <tool-name>      : see BenchKit documentation, the name of the tool
     <input>          : see BenchKit documentation, the name of the directory where the tool is executed

IMPORTANT: you must run ./BenchKitStart.sh in the directory you unpack the distribution.
IMPORTANT: in this version, only one core is allocated to the VM. This can be changed in the file vm.sh
           (at least for Qemu).
```

By default, only one core is allocated. To change this (QEMU only), have a look in the `launch_a_vm_with_qemu_or_vbox` in file `vm.sh`:

```
   $KVM -vnc :$VNC \
         -enable-kvm\
         -smp 1\
         -cpu host \
         -daemonize \
         -k $KEYBOARD \
         -m $MAXMEM \
         -drive file=$HDD \
         -net nic,vlan=1 -net user,vlan=1 -name MCC \
         -redir tcp:$SSHP::22
```

should be replaced by (we consider here 4 cores, only the `-smp` parameter value is changed)

```
   $KVM -vnc :$VNC \
         -enable-kvm\
         -smp 4\
         -cpu host \
         -daemonize \
         -k $KEYBOARD \
         -m $MAXMEM \
         -drive file=$HDD \
         -net nic,vlan=1 -net user,vlan=1 -name MCC \
         -redir tcp:$SSHP::22
```

We have no idea on the way to change this with VirtualBox.

Please remind that the monitoring functions of ***BenchKit*** have been disabled to remove delicate dependencies and ease the installation on your target machine.