PyB Manual

Last update of this file: 03.10.2015 by John Witulski
E-mail: John.Witulski@hhu.de

Index:

1. What is PyB

PyB is a B implementation written in python. More informations about B: http://en.wikipedia.org/wiki/B-Method

It requires java and python 2.7

2. Use PyB

checking mode:
No animation. Evaluate properties, assertions and invariant clause in the first state. Missing cluases are omitted.
python pyB.py **-c** <Filename>

read eval print mode:
evaluate prediactes and expressions in repl.
python pyB.py **-repl**

animation mode:
Interactive step by step execution of operations and statements. Set up constants and initialization are separate steps.
python pyB.py <FileName>

evaluate given state (second tool chain). Executed set up constants and initialization using values from solution file.
python pyB.py **-c** <FileName> <SolutionFilename>

experimental stand alone model checking:
python pyB.py **-mc** <FileName>

be sure your PYTHONPATH is set.
Single predicates and expression files need #PREDICATE / #EXPRESSION at the first line

3. Flags and Switches

modify pyB/config.py:

3.1 ENABLE_ASSERTIONS:
values: True, False.
summary: evaluate assertions clause (e.g. in checking mode)

3.2 BMACHINE_SEARCH_DIR:
values: any file path.
summary: search dir of included, extended, used and seen machines

3.3 PROPERTIES_TIMEOUT:
values: double greater than zero
summary: timeout for a subpredicate eval. In properties clause

4. Modify PyB

Parser modification:
1. Check out the gradle build tool from the gradle from http://gradle.org
2. Modify Main.java and ASTPython.java and move them in the subfolder src/main/java/pyB
3. Run build.gradle file from the project root. On mac os:

$ **PATH=$PATH:<YOUR GRADLE PATH> gradle uberjar**

The output looks like this:

:compileJava UP-TO-DATE
:processResources UP-TO-DATE
:uberjar UP-TO-DATE

BUILD SUCCESSFUL

Total time: 4.737 secs

4. A File javaparser.jar has been created in /build/libs/.
5. The Java Parser can be tested on mac os by

$ **java -jar build/libs/javaparser.jar input.txt**

input.txt must contain a B-machine or a B-predicate / B- expression beginning with #PREDICATE or #EXPRESSION. E.g #PREDICATE 1<2

6. Move the jar file to <PYB PATH>/jars/ to use it with pyB or modify the path in config.py

C translation using PyPy tool chain:
$ python ../pypy/rpython/translator/goal/translate.py <File>

5. Supported Syntax

More informations at ClearSY - B LANGUAGE REFERENCE MANUAL (VERSION 1.8.6)

url: http://www.math.pku.edu.cn/teachers/qiuzy/fm_B/Atelier_B/B-manrefb1.8.6.uk.pdf

Python-Version:

propositions, quantified predicates, equality predicates, inclusion predicates, integer comparison predicates, boolean expressions, arithmetic operators, maplets, set operators, default sets (e.g NAT, BOOL), set operators, records, structs, relations, relation operators, iteration and transitive closure*, functions, function operators, sequences, sequence operators, substitutions, clauses

missing features:
(?) $0 specific value(5.1 page 38)
(?) character string
trees (5.20. page 72)
values clause
refinements, implementations
Event-B

* implementation similar to ProB

C-Version:

More informations at tests/test_pypy_translation_objects.py

simple arithmetic, integer comparison predicates

6. Translate to C using PyPy

1. Switch USE_RPYTHON_CODE and USE_RPYTHON_POPEN flags in config.py to True.
2. Install pypy in PYPYDIR
3. execute on console: PYTHONPATH=<PYPY_DIR>:. python ../pypy/rpython/translator/goal/translate.py --batch pyB_Rpython.py
   from pyB root with PYPYDIR = pypy install folder
4. execute on console: ./pyB_Rpython-c <BMACHINE>
   from pyB root with BMACHINE = path to B machine file

~ 402.0 s seconds needed

Feature list:

	PyB-C	PyB-Python
Predicate Unit	partial	yes
Expression Unit	no	partial
contraint clause	yes	yes
properties clause	no	yes
invariant clause	yes	yes
assertion clause	no	yes
conjunct predicate	yes	yes
disjunct predicate	yes	yes
impl. Predicate	yes	yes
equivalence predicate	yes	yes
negation predicate	yes	yes
for all predicate	naive	using constraints
exists predicate	naive	using constraints
equal predicate	yes	yes
not equal predicate	yes	yes
set extention	yes	yes
empty set	yes	yes
comprehension set	no	using constraints
intersection	symbolic	symbolic
union	symbolic	symbolic
couple	naive	naive
powerset	symbolic	symbolic
powerset1	symbolic	symbolic
card expression	yes	yes
general union	naive	naive
general intersection	naive	naive
qu. intersection	partial	symbolic
qu. union	partial	symbolic
member predicate	partial	yes
not member predicate	partial	yes
subset predicate	partial	yes
not subset predicate	partial	yes
strict subset	partial	yes
not strict subset	partial	yes

	PyB-C	PyB-Python
NAT	symbolic	symbolic
NAT1	symbolic	symbolic
INT	symbolic	symbolic
NATURAL	symbolic	symbolic
NATUARAL1	symbolic	symbolic
INTEGER	symbolic	symbolic
min	yes	yes
max	yes	yes
add	yes	yes
Minus / set sub	yes	yes
Mult / cart	yes	yes
div	yes	yes
modulo	yes	yes
power of	yes	yes
interval	naive	symbolic
gen. Sum	naive	using constraints
gen prod	naive	using constraints
greater	yes	yes
less	yes	yes
greater equal	yes	yes
less equal	yes	yes
relation	symbolic	symbolic
domain	partial	yes
range	partial	yes
composition	partial	yes
identity expression	symbolic	symbolic
domain restriction	partial	partial
domain subtraction	partial	partial
range restriction	partial	partial
range subtraction	partial	partial
invserse	symbolic	symbolic
image	partial	yes
overwrite	partial	yes
direct prod	partial	yes
parallel prod	partial	yes
iteration	naive	naive
closure	naive	naive
ref. Closure	naive	naive
first proj	partial	yes
second proj	partial	yes

	PyB-C	PyB-Python
part function	symbolic	symbolic
total function	symbolic	symbolic
part. Inj function	symbolic	symbolic
total inj. Function	symbolic	symbolic
part. Surj. Function	symbolic	symbolic
total surj. Function	symbolic	symbolic
total bij. Function	symbolic	symbolic
part. Bij. Function	no	symbolic
lambda expression	no	symbolic
function expression	yes	yes
empty sequ	yes	yes
seq expression	symbolic	symbolic
Seq 1 expression	symbolic	symbolic
iseq	symbolic	symbolic
iseq1	symbolic	symbolic
perm	symbolic	symbolic
concat	yes	yes
insert	yes	yes
Instert tail	yes	yes
seq seq expression	yes	yes
size	yes	yes
rev	yes	yes
restrict	yes	yes
restrict tail	yes	yes
first	yes	yes
last	yes	yes
tail	yes	yes
front	yes	yes
gen conc	no	yes

	PyB-C	PyB-Python
string	yes	yes
bool	yes	yes
uni minus	yes	yes
int	yes	yes
min_int	yes	yes
max_int	yes	yes
id	yes	yes
prim id	no	partial
bool set	yes	yes
true	yes	yes
false	yes	yes
struct	yes	yes
rec	yes	yes
record	yes	yes
string set	yes	yes
trans	yes	yes
func	yes	yes
external func	no	yes
skip	yes	yes
:=	partial	yes
:{P}	yes	yes
::	yes	yes
S1 || S2	yes	yes
BEGIN s END	yes	yes
S1 ; S2	yes	yes
WHILE	yes	yes
PRE	yes	yes
ASSERT	yes	yes
IF	yes	yes
CHOICE	yes	yes
SELECT	yes	yes
CASE	yes	yes
VAR	yes	yes
ANY	yes	yes
OP	yes	yes