

Concurrency

EECS 4315

www.eecs.yorku.ca/course/4315/

The readers-writers problem

The readers and writers problem, due to Courtois, Heymans and Parnas, is a classical concurrency problem. It models access to a database. There are many competing threads wishing to read from and write to the database. It is acceptable to have multiple threads reading at the same time, but if one thread is writing then no other thread may either read or write. A thread can only write if no thread is reading.

```
public class Database {
    private boolean writing;
    private int readers;
    ...
    public void read() {
        this.beginRead();
        // read
        this.endRead();
    }
}
```

Add `assert !this.writing` in the `read` method where the database is read. If the assertion fails, an exception is thrown. JPF detects exceptions that are thrown and not caught.

Configuring the report

```
sourcepath=<path to ReadersAndWriters.java>
```

```
report.class=gov.nasa.jpf.report.Reporter
```

```
report.publisher=console
```

```
report.console.property_violation=error,trace
```

```
report.console.show_steps=true
```

```
report.console.show_method=true
```

```
report.console.show_code=true
```

```
JavaPathfinder core system v8.0 (rev 32+) - (C) 2005-2014 U
===== search
===== error
gov.nasa.jpfcvm.NoUncaughtExceptionsProperty
java.lang.AssertionError
at concurrency.Database.read(concurrency/Database.java:30)
at concurrency.Reader.run(concurrency/Reader.java:25)
===== trace
----- trace
gov.nasa.jpfcvm.choice.ThreadChoiceFromSet {id:"ROOT" ,1/1
  java.lang.Boolean.<clinit>
    invokestatic java.lang.Boolean.<clinit>()V
  java.lang.Boolean.<clinit>()V
    new java.lang.Boolean@bc
    dup
    iconst_1
    invokespecial java.lang.Boolean.<init>(Z)V
  java.lang.Boolean.<init>(Z)V
```

The extension jpf-shell can be downloaded from the Mercurial repository

<http://babelfish.arc.nasa.gov/hg/jpf/jpf-shell>.

How to install an extension of JPF is described in the Section 1.9 of the notes.

Site properties file:

```
# JPF site configuration

jpf-core=<path to jpf-core>
jpf-shell=<path to jpf-shell>

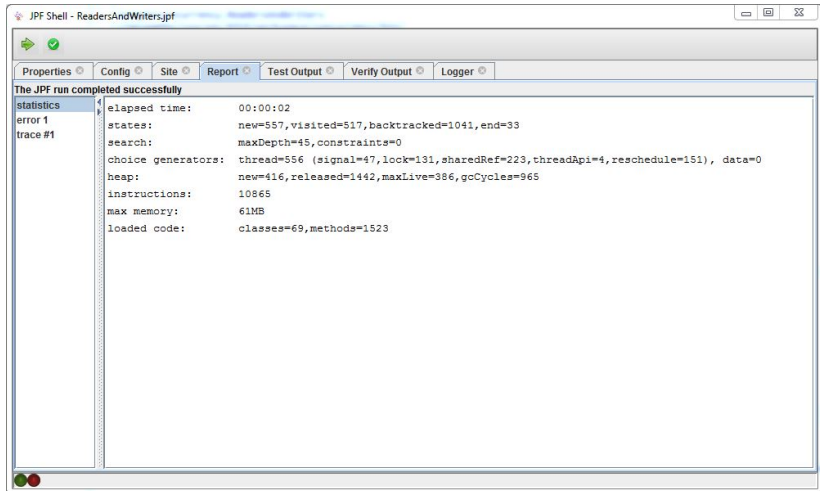
extensions=${jpf-core}
```

Configuration file:

```
target=ReadersAndWriters
classpath=<path to ReadersAndWriters.class>
sourcepath=<path to ReadersAndWriters.java>

@using=jpf-shell
shell=.shell.basicshell.BasicShell
```

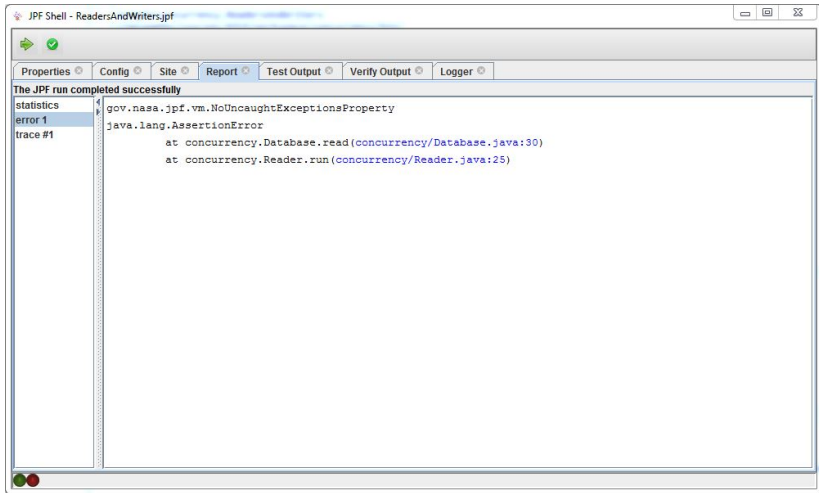
Report of jpf-shell



The screenshot shows a window titled "JPF Shell - ReadersAndWriters.jpf". The interface includes a menu bar with options: Properties, Config, Site, Report (selected), Test Output, Verify Output, and Logger. Below the menu bar, a status bar indicates "The JPF run completed successfully". A sidebar on the left contains a list of items: statistics (selected), error 1, and trace #1. The main content area displays the following report data:

```
elapsed time:      00:00:02
states:           new=557,visited=517,backtracked=1041,end=33
search:          maxDepth=45,constraints=0
choice generators: thread=556 (signal=47,lock=131,sharedRef=223,threadApi=4,reschedule=151), data=0
heap:            new=416,released=1442,maxLive=386,gcCycles=965
instructions:    10865
max memory:      61MB
loaded code:     classes=69,methods=1523
```

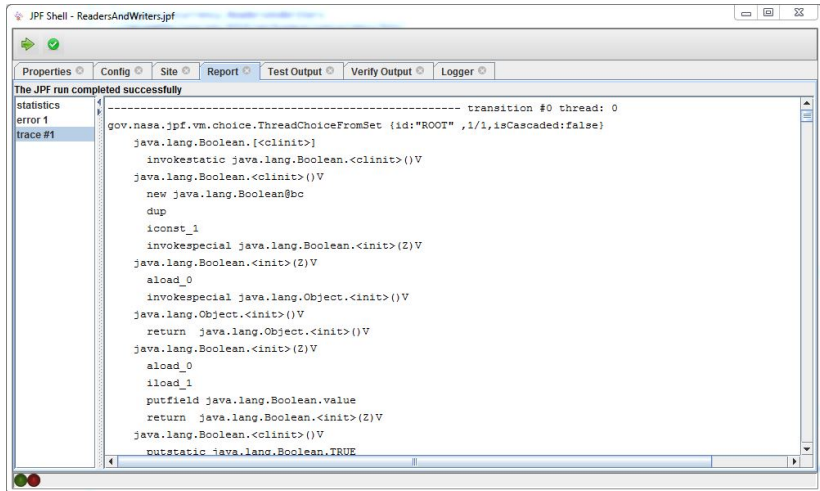

Report of jpf-shell



The screenshot shows a window titled "JPF Shell - ReadersAndWriters.jpf". The window has a menu bar with "Properties", "Config", "Site", "Report", "Test Output", "Verify Output", and "Logger". Below the menu bar, a status bar indicates "The JPF run completed successfully". The main content area displays a stack trace for an error:

```
gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
error 1
trace #1
    at concurrency.Database.read (concurrency/Database.java:30)
    at concurrency.Reader.run (concurrency/Reader.java:25)
```

Report of jpf-shell



```
JPF Shell - ReadersAndWriters.jpf
Properties Config Site Report Test Output Verify Output Logger
The JPF run completed successfully
statistics ----- transition #0 thread: 0
error 1 gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"ROOT" ,1/1,isCascaded:false}
trace #1
  java.lang.Boolean.<clinit>
    invokestatic java.lang.Boolean.<clinit>()V
  java.lang.Boolean.<clinit>()V
    new java.lang.Boolean@bc
    dup
    iconst_1
    invokespecial java.lang.Boolean.<init>(Z)V
  java.lang.Boolean.<init>(Z)V
    aload_0
    invokespecial java.lang.Object.<init>()V
  java.lang.Object.<init>()V
    return java.lang.Object.<init>()V
  java.lang.Boolean.<init>(Z)V
    aload_0
    iload_1
    putfield java.lang.Boolean.value
    return java.lang.Boolean.<init>(Z)V
  java.lang.Boolean.<clinit>()V
    putstatic java.lang.Boolean.TRUE
```

Cyrille Artho and Qiyi Tang. JPF Visual. Presented at the JPF Workshop, Urbana-Champaign, IL, USA, November 2017.

The extension `jpf-visual` can be downloaded from the BitBucket repository <https://bitbucket.org/qiyitang71/jpf-visual>.

How to install an extension of JPF is described in the Section 1.9 of the notes.

Site properties file:

```
# JPF site configuration

jpf-core=<path to jpf-core>
jpf-shell=<path to jpf-shell>
jpf-visual=<path to jpf-visual>

extensions=${jpf-core}
```

Configuration file:

```
target=ReadersAndWriters
classpath=<path to ReadersAndWriters.class>
sourcepath=<path to ReadersAndWriters.java>

@using=jpf-visual
shell=.shell.basicshell.BasicShell
shell.panels+=,errorTrace
shell.panels.errorTrace=ErrorTracePanel
report.publisher+=,errorTracePrinter
report.errorTracePrinter.class=ErrorTracePrinter
report.errorTracePrinter.property_violation=trace
```

Report of jpf-visual

The JPF run completed successfully

Properties | Config | Site | Report | Test Output | Verify Output | Logger | Error Trace

Trans. | main 0 | Thread-1 1 | Thread-2 2 | Thread-3 3 | Thread-4 4

Trans.	main 0	Thread-1 1	Thread-2 2	Thread-3 3	Thread-4 4
0-4	final int READERS = 2; ... public class ReadersAndWriters {				
5-7	package concurrency; ... }				
8	package concurrency; ... this.beginRead();				
9-15	package concurrency; ... }				
16-17	this.beginRead(); ... this.wait();				
	}				

Thread State

Custom filter...

wait/notify
thread start/join
(un)lock: concurrency.Datab...

Report of jpf-visual

JPF Shell - ReadersAndWriters.jpf

Properties Config Site Report Test Output Verify Output Logger Error Trace

The JPF run completed successfully

Collapse all
Expand all
 wait/notify
 thread start/join
 (un)lock: concurrency.Datab...
Custom filter...

Trans.	main 0	Thread-1 1	Thread-2 2	Thread-3 3	Thread-4 4
0-4	<pre>final int READERS = 2; ... public class ReadersAndWriters {</pre>				
5-7		<pre>package concurrency; ... }</pre>			
8			<pre>package concurrency; ... this.beginRead();</pre>		
9-15				<pre>package concurrency; ... }</pre>	
16-17			<pre>this.beginRead(); ... this.wait();</pre>		
					<pre>}</pre>

Outline

0-4
5-7
8

Report of jpf-visual

JPF Shell - ReadersAndWriters.jpf

Properties | Config | Site | Report | Test Output | Verify Output | Logger | Error Trace

The JPF run completed successfully

wait/notify

thread start/join

(un)lock: concurrency.Datab...

Custom filter...

field: concurrency.Database...

Trans.	main 0	Thread-1 1	Thread-2 2	Thread-3 3	Thread-4 4
0-4	<pre>final int READERS = 2; ... this.writing = false; ... public class ReadersAndWriters {</pre>				
5-7	<pre>package concurrency; ... if (this.writing) { ... assert !this.writing; ... }</pre>				
8	<pre>package concurrency; ... this.beginRead();</pre>				
	<pre>package concurrency; ... </pre>				

Thread State

Report of jpf-visual

JPF Shell - ReadersAndWriters.jpf

Properties | Config | Site | Report | Test Output | Verify Output | Logger | Error Trace

The JPF run completed successfully

wait/notify
 thread start/join
 (un)lock: concurrency.Datab...
Custom filter...
 field: concurrency.Database...

Trans.	main 0	Thread-1 1	Thread-2 2	Thread-3 3	Thread-4 4
0-4	<pre>final int READERS = 2; ... this.writing = false; ... public class ReadersAndWriters {</pre>				
5-7	<pre>package concurrency; ... if (this.writing) { ... assert !this.writing; ... this.notifyAll(); }</pre>				
8	<pre>package concurrency; ... this.beginRead();</pre>				
	<pre>package concurrency;</pre>				

Thread State

- `Popper` class
- `Pusher` class
- `Stack` class
- `Main` class

Dining savages problem

A tribe of savages eats communal dinners from a large pot that can hold M servings of stewed missionary. When savages want to eat, they help themselves from the pot, unless it is empty. If the pot is empty, the savage wakes up the cook and then waits until the cook has refilled the pot.