

Assignment 13

- 氏名: 栗林健太郎
- 学生番号: 2030006
- 作成日: 2020年11月30日

SimpInc

まずは競合の起こらないSimpInc.javaを実行する。

```
● ● ● て32 fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13
antipop@PMAC670S ~ / s / g / k / j / assignment13 (master) [1] > javac SimpInc.java ; java SimpInc
count: 1
antipop@PMAC670S ~ / s / g / k / j / assignment13 (master) > javac SimpInc.java ; java -enableassertions SimpInc
count: 1
Exception in thread "main" java.lang.AssertionError
    at SimpInc.main(SimpInc.java:21)
antipop@PMAC670S ~ / s / g / k / j / assignment13 (master) [1] > █
```

上記の通り、count: 1が表示される（2度めの実行のように–enableassertionsオプションをつけて実行するとアサーションエラーが起こる）。

SimpIncをJPFで検査する。

```

● ● ● ˇ⌘2          fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13
SimpInc.java:21      : assert count == 2;
[21 insn w/o sources]

===== snapshot #1
thread java.lang.Thread:{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
call stack:
    at SimpInc.main(SimpInc.java:21)

=====
results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty "java.lang.AssertionError  at SimpInc.main(SimpInc...."

=====
statistics
elapsed time: 00:00:00
states: new=4,visited=0,backtracked=0,end=0
search: maxDepth=4,constraints=0
choice generators: thread=4 (signal=0,lock=1,sharedRef=0,threadApi=2,reschedule=1), data=0
heap: new=383,released=5,maxLive=356,gcCycles=3
instructions: 3394
max memory: 245MB
loaded code: classes=66,methods=1516

===== search finished: 20/11/30 23:28
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> █

```

以下の通り、アサーションエラーで終了する。

```

===== snapshot #1
thread java.lang.Thread:
{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspe
ndCount:0}
call stack:
    at SimpInc.main(SimpInc.java:21)

=====
results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
"java.lang.AssertionError  at SimpInc.main(SimpInc...."

```

スレッドの実行の様子は以下の通りである（競合に関係する箇所のみ抜粋）。

```

===== trace #1
----- transition #1
thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"START"
,1/2,isCascaded:false}
    [2 insn w/o sources]
        SimpInc.java:12           : count++;
        SimpInc.java:13           : t.join();
    [1 insn w/o sources]
----- transition #2
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"JOIN"
,1/1,isCascaded:false}
    [1 insn w/o sources]

```

```
SimpInc.java:1          : public class SimpInc extends Thread {  
SimpInc.java:6          :     count2++;  
SimpInc.java:7          : }  
SimpInc.java:4          :  
:  
:
```

count変数に対する操作はthread: 0のみで行われている。そのため、thread: 0（メインスレッド）とthread: 1の間でcount変数にたいする競合は起こっていない。

SimpConcInc

競合の起こるSimpConcInc.javaを実行する。

```
● ● ● ~%2  
fish /Users/antipop/sro/github.com/kentaro/jaist-i219-assignment/assignment13  
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac SimpConcInc.java ; java SimpConcInc  
count: 2  
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac SimpConcInc.java ; java -enableassertions SimpConcInc  
count: 2  
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> █
```

上記の通り、count: 2が表示される。-enableassertionsオプションをつけて実行してもエラーが起こらないため、一見すると正しく実行されているかのように見える。

SimpConcIncをJPFで検査する。

```

● ○ ● ˇ⌘2          fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13
SimpConcInc.java:20      : assert count == 2;
[21 insn w/o sources]

===== snapshot #1
thread java.lang.Thread:{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
call stack:
    at SimpConcInc.main(SimpConcInc.java:20)

===== results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty "java.lang.AssertionError at SimpConcInc.main(Simp..."

===== statistics
elapsed time: 00:00:00
states: new=12,visited=1,backtracked=5,end=1
search: maxDepth=8,constraints=0
choice generators: thread=11 (signal=0,lock=1,sharedRef=4,threadApi=4,reschedule=2), data=0
heap: new=389,released=22,maxLive=356,gcCycles=7
instructions: 3452
max memory: 245MB
loaded code: classes=66,methods=1516

===== search finished: 20/11/30 23:38
antipop@PMAC670S ~$/g/k/j/assignment13 (master)> █

```

以下の通り、アサーションエラーで終了する。

```

===== snapshot #1
thread java.lang.Thread:
{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspe
ndCount:0}
call stack:
    at SimpConcInc.main(SimpConcInc.java:20)

===== results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
"java.lang.AssertionError at SimpConcInc.main(Simp..."


```

スレッドの実行の様子は以下の通りである（競合に関係する箇所のみ抜粋）。

```

===== trace #1
----- transition #1
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"START"
,2/2,isCascaded:false}
    [1 insn w/o sources]
    SimpConcInc.java:1           : public class SimpConcInc extends Thread
{
    SimpConcInc.java:5           : count++;
----- transition #2
thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,1/2,isCascaded:false}
    [2 insn w/o sources]

```

```

SimpConcInc.java:11          : count++;
----- transition #3
thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,1/2,isCascaded:false}
SimpConcInc.java:11          : count++;
----- transition #4
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,2/2,isCascaded:false}
SimpConcInc.java:5          : count++;
----- transition #5
thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,1/2,isCascaded:false}
SimpConcInc.java:11          : count++;
SimpConcInc.java:12          : t.join();
[1 insn w/o sources]
----- transition #6
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"JOIN"
,1/1,isCascaded:false}
SimpConcInc.java:5          : count++;
SimpConcInc.java:6          : }
SimpConcInc.java:4          : public void run() {

```

ふたつのスレッドによる各transitionのフローは以下の通りである（競合に関係する箇所のみ抜粋）。

- transition #1: **thread: 1**が**count**から**0**を読み出し
- transition #2: **thread: 0**が**count**から**0**を読み出し
- transition #3: **thread: 0**が**count**をインクリメント
- transition #4: **thread: 1**が**count**をインクリメント
- transition #5: **thread: 0**が**count**に**1**を書き込み
- transition #6: **thread: 1**が**count**に**1**を書き込み

thread: 0（メインスレッド）と**thread: 1**との間で**count**変数に対する競合が起こっている。結果としてアサーション時には**count**の値は**1**になるため、エラーが発生する。

UnsafeInc

カウンターのクラス**FCounter**を2つのスレッドで共有する**UnsafeInc**を実行する。

```
● ○ ●  2 fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13  
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac UnsafeInc.java ; java UnsafeInc  
count: 2  
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac UnsafeInc.java ; java -enableassertions UnsafeInc  
count: 2  
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> █
```

上記の通り、**count: 2**が表示される。**-enableassertions**オプションをつけて実行してもエラーが起こらないため、一見すると正しく実行されているかのように見える。

UnsafeIncをJPFで検査する。

```
● ○ ●  2 fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13  
UnsafeInc.java:15 : assert FCounter.get() == 2;  
[21 insn w/o sources]  
===== snapshot #1  
thread java.lang.Thread:{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}  
call stack:  
    at UnsafeInc.main(UnsafeInc.java:15)  
  
===== results  
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty "java.lang.AssertionError" at UnsafeInc.main(Unsafe...)"  
  
===== statistics  
elapsed time: 00:00:00  
states: new=26,visited=6,backtracked=17,end=2  
search: maxDepth=15,constraints=0  
choice generators: thread=25 (signal=0,lock=5,sharedRef=5,threadApi=9,reschedule=6), data=0  
heap: new=409,released=61,maxLive=365,gcCycles=25  
instructions: 3772  
max memory: 245MB  
loaded code: classes=67,methods=1520  
  
===== search finished: 20/11/30 23:59  
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> █
```

以下の通り、アサーションエラーで終了する。

```
===== snapshot #1  
thread java.lang.Thread:  
{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspe  
ndCount:0}  
call stack:  
    at UnsafeInc.main(UnsafeInc.java:15)
```

```
===== results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
"java.lang.AssertionError at UnsafeInc.main(Unsafe...)"
```

ふたつのスレッドによる各transitionのフローは以下の通りである（競合に関係する箇所のみ抜粋）。

```
===== trace #1
----- transition #7
thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK"
,2/2,isCascaded:false}
UnsafeInc.java:3 : (new FCounter()).inc();
FCounter.java:15 : x++;
----- transition #8
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,1/2,isCascaded:false}
UnsafeInc.java:3 : (new FCounter()).inc();
FCounter.java:15 : x++;
----- transition #9
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,1/2,isCascaded:false}
FCounter.java:15 : x++;
----- transition #10
thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,2/2,isCascaded:false}
FCounter.java:15 : x++;
----- transition #11
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_CLASS"
,1/2,isCascaded:false}
FCounter.java:15 : x++;
FCounter.java:16 : }
----- transition #13
thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"JOIN"
,1/1,isCascaded:false}
FCounter.java:15 : x++;
FCounter.java:16 : }
```

ふたつのスレッドによる各transitionのフローは以下の通りである。

- transition #7: **thread: 2**が`x`から`0`を読み出し
- transition #8: **thread: 1**が`x`から`0`を読み出し
- transition #9: **thread: 1**が`x`をインクリメント
- transition #10: **thread: 2**が`x`をインクリメント

- transition #11: **thread: 1**が**x**に**1**を書き込み
- transition #13: **thread: 2**が**x**に**1**を書き込み

thread: 1と**thread: 2**との間で**FCounter**クラスのスタティック変数**x**に対する競合が起こっている。結果としてアサーション時には**count**の値は**1**になるため、エラーが発生する。

UnsafeIncでは、スレッドが**FCounter**オブジェクトをそれぞれ作り、それぞれのオブジェクトに対してロックを取っているため、**FCounter**のスタティック変数**x**に対してはインクリメント処理が競合してしまうためである。

SafeInc

カウンターのクラス**GCounter**を2つのスレッドで共有する**SafeInc**を実行する。

```
● ● ● 末端2
fish /Users/antipop/src/github.com/kentaro/jaist-l219-assignment/assignment13
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac SafeInc.java ; java SafeInc
count: 2
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac SafeInc.java ; java -enableassertions SafeInc
count: 2
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> █
```

上記の通り、**count: 2**が表示される。

次に、**SafeInc**をJPFで検査する。

```
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13

=====
SafeInc.main()
=====
===== system under test
count: 2
count: 2
count: 2

=====
results
no errors detected

=====
statistics
elapsed time: 00:00:00
states: new=166,visited=179,backtracked=345,end=3
search: maxDepth=16,constraints=0
choice generators: thread=166 (signal=0,lock=49,sharedRef=61,threadApi=15,reschedule=41), data=0
heap: new=467,released=137,maxLive=366,gcCycles=304
instructions: 5847
max memory: 245MB
loaded code: classes=63,methods=1479

=====
search finished: 20/12/01 0:44
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

以下の通り、エラーなく終了する。

```
=====
results
no errors detected
```

FCounterと違い、ふたつのスレッドそれぞれが**GCounter**のスタティック変数**lock**で示されるオブジェクトに対してロックを取った上でインクリメントを実行するため、同期が適切に行われる。

SafeInc2

カウンターのクラス**Counter**を2つのスレッドで共有する**SafeInc2**を実行する。

```
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13

antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac SafeInc2.java ; java SafeInc2
count: 2
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac SafeInc2.java ; java -enableassertions SafeInc2
count: 2
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

上記の通り、**count: 2**が表示される。

SafeInc2をJPFで検査する。

```
fish /Users/antipop/src/github.com/kentaro/aiast-i219-assignment/assignment13

=====
system under test
SafeInc2.main()

=====
search started: 20/12/01 0:43

count: 2
count: 2
count: 2

=====
results
no errors detected

=====
statistics
elapsed time: 00:00:00
states: new=105,visited=104,backtracked=209,end=3
search: maxDepth=14,constraints=0
choice generators: thread=105 (signal=0,lock=20,sharedRef=46,threadApi=12,reschedule=27), data=0
heap: new=390,released=71,maxLive=364,gcCycles=162
instructions: 4517
max memory: 245MB
loaded code: classes=63,methods=1478

=====
search finished: 20/12/01 0:43
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> 
```

以下の通り、エラーなく終了する。

```
=====
results
no errors detected
```

ふたつのスレッドそれぞれが、スタティック変数**counter**として宣言された**Counter**オブジェクトを共有しているが、**Counter**クラスでは変数**x**がスタティックではなく、かつ、インクリメントする**inc**メソッドが同期されている。それぞれのスレッドは**counter**オブジェクトに対してロックを取った上でインクリメントを実行するため、同期が適切に行われる。

Bounded Buffer Problem

前回実装したBounded Buffer Problemを実装したクラス群と**BBProb**を実行する。

```
● ● ● TextEdit  
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac BBProb.java ; java BBProb  
msgsSent: [0, 1]  
msgsReceived: [0, 1]  
Success!  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac BBProb.java ; java -enableassertions BBProb  
msgsSent: [0, 1]  
msgsReceived: [0, 1]  
Success!  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

上記の通り、問題なく処理が成功して終了する。

BBProbをJPFで検査する。

```
● ● ● TextEdit  
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13  
msgsReceived: [0, 1]  
Success!  
msgsSent: [0, 1]  
msgsReceived: [0, 1]  
Success!  
msgsSent: [0, 1]  
msgsReceived: [0, 1]  
Success!  
  
===== results  
no errors detected  
  
===== statistics  
elapsed time: 00:00:00  
states: new=1797,visited=2361,backtracked=4158,end=3  
search: maxDepth=87,constraints=0  
choice generators: thread=1797 (signal=43,lock=137,sharedRef=1473,threadApi=90,reschedule=54), data=0  
heap: new=1101,released=344,maxLive=675,gcCycles=3752  
instructions: 42141  
max memory: 245MB  
loaded code: classes=80,methods=1835  
  
===== search finished: 20/12/01 1:03  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

以下の通り、エラーなく終了する。

```
===== results  
no errors detected
```

次に、送信されたメッセージを **log** 変数に記録しておくように改修した **MonitorBBuf2** をバッファとして使うようにした **Sender2** および **Receiver2** を用いて、メッセージを送る **Sender2** を2つにした **BBProb2** を実行する。

```
● ● ● TextEdit  
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac BBProb2.java ; java BBProb2  
msgsSent: [0, 1]  
msgsReceived: [0, 1, 0, 1]  
Failure!  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac BBProb2.java ; java -enableassertions BBProb2  
msgsSent: [0, 1]  
msgsReceived: [0, 1, 0, 1]  
Failure!  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

上記の通り、問題なく処理が成功して終了する。

BBProb2をJPFで検査する。

```
● ● ● TextEdit  
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13  
msgsReceived: [0, 0, 1, 1]  
Failure!  
msgsSent: [0, 1]  
msgsReceived: [0, 1, 0, 1]  
Failure!  
msgsSent: [0, 1]  
msgsReceived: [0, 0, 1, 1]  
Failure!  
===== results  
no errors detected  
===== statistics  
elapsed time: 00:00:32  
states: new=164735,visited=358520,backtracked=523255,end=8  
search: maxDepth=232,constraints=0  
choice generators: thread=164735 (signal=2519,lock=11913,sharedRef=146160,threadApi=607,reschedule=3536), data=0  
heap: new=13571,released=7996,maxLive=682,gcCycles=487529  
instructions: 3163980  
max memory: 501MB  
loaded code: classes=80,methods=1835  
===== search finished: 20/12/01 1:19  
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

以下の通り、エラーなく終了する。

```
===== results  
no errors detected
```

さらに、上記とほぼ変わらないが、メッセージをputする際にwhileで待つところをifに変更したFMonitorBBuf1をバッファとして用いるSender/Receiverを使うFBBProb1を実行する。

```
● ● ● TextEdit
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac FBBProb1.java ; java FBBProb1
msgsSent: [0, 1]
msgsReceived: [0, 1, 0, 1]
Failure!
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> javac FBBProb1.java ; java -enableassertions FBBProb1
msgsSent: [0, 1]
msgsReceived: [0, 1, 0, 1]
Failure!
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

上記の通り、メッセージを適切に送受信することに失敗している。

FBBProb1をJPFで検査する。

```
● ● ● TextEdit
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13
thread FReceiver1:{id:3,name:Thread-3,status:WAITING,priority:5,isDaemon:false,lockCount:1,suspendCount:0}
  waiting on: FMonitorBBuf1@179
  call stack:
    at java.lang.Object.wait(Object.java)
    at FMonitorBBuf1.get(FMonitorBBuf1.java:36)
    at FReceiver1.run(FReceiver1.java:23)

=====
          results
error #1: gov.nasa.jpf.vm.NotDeadlockedProperty "deadlock encountered:  thread java.lang.Thread:{i...}"

=====
          statistics
elapsed time:      00:00:03
states:           new=14787,visited=24545,backtracked=39217,end=4
search:            maxDepth=220,constraints=0
choice generators: thread=14786 (signal=212,lock=1000,sharedRef=13046,threadApi=235,reschedule=293), data=0
heap:              new=1745,released=1054,maxLive=682,gcCycles=36693
instructions:     257397
max memory:       501MB
loaded code:       classes=80,methods=1835

=====
          search finished: 20/12/01 1:35
antipop@PMAC670S ~$ g/k/j/assignment13 (master)> █
```

以下の通り、デッドロックを検出する。

```
=====
          results
error #1: gov.nasa.jpf.vm.NotDeadlockedProperty "deadlock encountered:
thread java.lang.Thread:{i...}"
```

3つのスレッドによる各transitionのフローは以下の通りである（競合に関係する箇所のみ抜粋）。

```

----- transition #72
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_OBJECT"
,1/3,isCascaded:false}
  FMonitorBBuf1.java:22      : if (noe >= capacity) {
  FMonitorBBuf1.java:23      : this.wait();
[1 insn w/o sources]
----- transition #82
thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_OBJECT"
,1/2,isCascaded:false}
  FMonitorBBuf1.java:22      : if (noe >= capacity) {
  FMonitorBBuf1.java:23      : this.wait();
[1 insn w/o sources]
----- transition #83
thread: 3
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"WAIT"
,1/1,isCascaded:false}
[1 insn w/o sources]
  FReceiver1.java:1          : /**
  FReceiver1.java:21         : for (int i = 0; i < nom; i++) {
  FReceiver1.java:23         : msgs.add(buf.get());
  FMonitorBBuf1.java:35      : while (noe <= 0) {
  FMonitorBBuf1.java:39      : if (noe > 0) {
  FMonitorBBuf1.java:40      : E e = queue.top();
  FMonitorBBuf1.java:41      : return head;
  NeQueue.java:25           : E e = queue.top();
  FMonitorBBuf1.java:40      : queue = queue.deq();
  FMonitorBBuf1.java:41      : return tail;
  FMonitorBBuf1.java:42      : queue = queue.deq();
  FMonitorBBuf1.java:43      : noe--;
  FMonitorBBuf1.java:43      : this.notifyAll();
[2 insn w/o sources]
  FMonitorBBuf1.java:44      : return e;
----- transition #85
thread: 1
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_OBJECT"
,1/2,isCascaded:false}
  FMonitorBBuf1.java:26      : if (noe < capacity) {
  FMonitorBBuf1.java:27      : queue = queue.enq(e);
----- transition #110
thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_OBJECT"
,1/2,isCascaded:false}
  FMonitorBBuf1.java:26      : if (noe < capacity) {
  FMonitorBBuf1.java:32      : }
  FSender1.java:24           : }
  FSender1.java:19           : for (int i = 0; i < msgs.size(); i++) {
----- transition #114
thread: 3
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"JOIN"
,1/1,isCascaded:false}
  FMonitorBBuf1.java:44      : return e;

```

```

FReceiver1.java:23          : msgs.add(buf.get());
[110 insn w/o sources]    :
FReceiver1.java:23          : msgs.add(buf.get());
FReceiver1.java:26          : }
FReceiver1.java:21          : for (int i = 0; i < nom; i++) {
FReceiver1.java:23          : msgs.add(buf.get());
FMonitorBBuf1.java:35       : while (noe <= 0) {
FMonitorBBuf1.java:39       : if (noe > 0) {
FMonitorBBuf1.java:40       : E e = queue.top();
NeQueue.java:25             : return head;
FMonitorBBuf1.java:40       : E e = queue.top();
FMonitorBBuf1.java:41       : queue = queue.deq();
NeQueue.java:21             : return tail;
FMonitorBBuf1.java:41       : queue = queue.deq();
FMonitorBBuf1.java:42       : noe--;
FMonitorBBuf1.java:43       : this.notifyAll();

[2 insn w/o sources]        :
FMonitorBBuf1.java:44       : return e;
FReceiver1.java:23          : msgs.add(buf.get());

[44 insn w/o sources]       :
FReceiver1.java:23          : msgs.add(buf.get());
FReceiver1.java:26          : }
FReceiver1.java:21          : for (int i = 0; i < nom; i++) {
FReceiver1.java:23          : msgs.add(buf.get());
FMonitorBBuf1.java:35       : while (noe <= 0) {
FMonitorBBuf1.java:39       : if (noe > 0) {
FMonitorBBuf1.java:40       : E e = queue.top();
NeQueue.java:25             : return head;
FMonitorBBuf1.java:40       : E e = queue.top();
FMonitorBBuf1.java:41       : queue = queue.deq();
NeQueue.java:21             : return tail;
FMonitorBBuf1.java:41       : queue = queue.deq();
FMonitorBBuf1.java:42       : noe--;
FMonitorBBuf1.java:43       : this.notifyAll();

[2 insn w/o sources]        :
FMonitorBBuf1.java:44       : return e;
FReceiver1.java:23          : msgs.add(buf.get());

[44 insn w/o sources]       :
FReceiver1.java:23          : msgs.add(buf.get());
FReceiver1.java:26          : }
FReceiver1.java:21          : for (int i = 0; i < nom; i++) {
FReceiver1.java:23          : msgs.add(buf.get());
FMonitorBBuf1.java:35       : while (noe <= 0) {
FMonitorBBuf1.java:36       : this.wait();

[1 insn w/o sources]

```

3つのスレッドによる各transitionのフローは以下の通りである。

- transition #72: **thread: 1 (sender1)** が**this.wait()**で待つ
- transition #82: **thread: 2 (sender2)** が**this.wait()**で待つ
- transition #83: **thread: 3 (receiver)** が**this.notifyAll()**を実行する
- transition #85: **thread: 1 (sender1)** がキューにメッセージを送信する

- transition #110: **thread: 2 (sender2)** はキューにメッセージを送信できない
- transition #13: **thread: 3 (receiver)** は**notifyAll()**を待ち続ける

receiverが**notifyAll()**を実行したあと、**sender1**がロックを取りメッセージを送信する。**sender2**はその後にロックを取るが、その際には**FMonitorBBuf1**の **if (noe < capacity) {** の箇所が**true**になることはなく、**sender2**はメッセージを送信することなく、**this.notifyAll()**を実行することはない。そのため、**this.wait()**で待ちに入った**receiver**スレッドが起きることはなくなり、デッドロックになってしまう。

最後に、**MonitorBBuf2**で、メッセージを**get**する際に**while**で待つ条件を**noe <= 0**から**noe < 0**に変更した**FMonitorBBuf2**をバッファとして用いるSender/Receiverを使う**FBBProb2**を実行する。

```
● ● ●  ~%2
fish /Users/antipop/src/github.com/kentaro/jaist-i219-assignment/assignment13
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac FBBProb2.java ; java FBBProb2
msgsSent: [0, 1]
msgsReceived: [0, 1, 0, 1]
Failure!
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> javac FBBProb2.java ; java -enableassertions FBBProb2
msgsSent: [0, 1]
msgsReceived: [0, 1, 0, 1]
Failure!
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> █
```

上記の通り、メッセージを適切に送受信することに失敗している。

FBBProb2をJPFで検査する。

```

[2 insn w/o sources]
FBBProb2.java:33      : System.out.println("msgsReceived: " + msgsReceived);
[429 insn w/o sources]
FBBProb2.java:33      : System.out.println("msgsReceived: " + msgsReceived);
[2 insn w/o sources]
FBBProb2.java:33      : System.out.println("msgsReceived: " + msgsReceived);
[2 insn w/o sources]
FBBProb2.java:35      : if (msgsReceived.equals(msgsSent)) {
[397 insn w/o sources]
FBBProb2.java:35      :   if (msgsReceived.equals(msgsSent)) {
FBBProb2.java:38      :     System.out.println("Failure!");
[2 insn w/o sources]
FBBProb2.java:41      :   assert msgsReceived.equals(log);
[611 insn w/o sources]
FBBProb2.java:41      :   assert msgsReceived.equals(log);
[21 insn w/o sources]

===== snapshot #1
thread java.lang.Thread:{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
call stack:
    at FBBProb2.main(FBBProb2.java:41)

===== results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty "java.lang.AssertionError" at FBBProb2.main(FBBProb...")

===== statistics
elapsed time: 00:00:00
states: new=969,visited=741,backtracked=1556,end=2
search: maxDepth=180,constraints=0
choice generators: thread=968 (signal=2,lock=48,sharedRef=776,threadApi=116,reschedule=26), data=0
heap: new=928,released=241,maxLive=682,gcCycles=1600
instructions: 29995
max memory: 245MB
loaded code: classes=84,methods=1876

===== search finished: 20/12/01 2:14
antipop@PMAC670S ~/s/g/k/j/assignment13 (master)> █

```

以下の通り、アサーションエラーで終了する。

```

===== snapshot #1
thread java.lang.Thread:
{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspe
ndCount:0}
call stack:
    at FBBProb2.main(FBBProb2.java:41)

===== results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
"java.lang.AssertionError" at FBBProb2.main(FBBProb...")


```

以下の通り、`while`で待つ条件が`0`未満となっているために`0`の際に待たなくなってしまうため、適切に処理が行われなくなってしまう。

```

----- transition #130
thread: 3
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK"
,2/2,isCascaded:false}
    FReceiver2.java:23          : msgs.add(buf.get());
    FMonitorBBuf2.java:35       : while (noe < 0) {


```

```
FMonitorBBuf2.java:39      : if (noe > 0) {  
FMonitorBBuf2.java:46      : return null;
```