

Estudo e desenvolvimento de novas funcionalidades do Java Pathfinder

MAC499 – Trabalho de Formatura Supervisionado

Orientadora: Ana Cristina Vieira de Melo (MAC-IME)

Rodrigo Della Vittoria Duarte

Bacharelado em Ciência da Computação
Instituto de Matemática e Estatística
Universidade de São Paulo

Introdução

- Alta demanda por software
- Orientação a Objetos largamente utilizado, especialmente Java
- Estrutura de Tratamento de Exceções vem ganhando espaço no desenvolvimento de novos software
- Poucas estratégias automáticas de testes e validação para sistemas orientados a objetos com tratamento de exceções

Problema

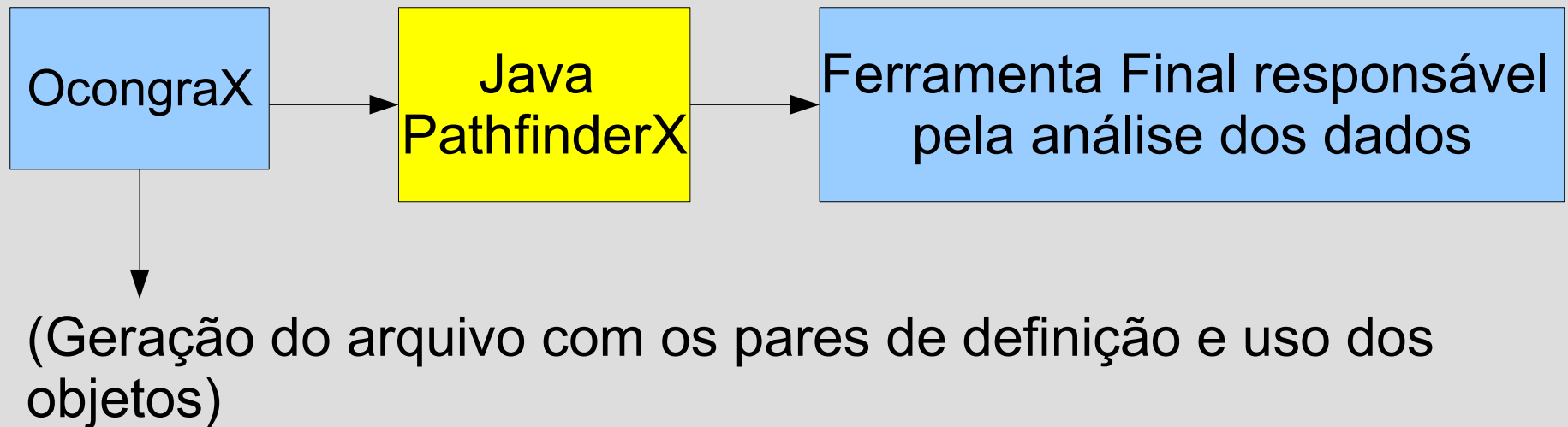
- A área de Testes e Validação consome muitos recursos
- Necessidade de escrever testes para todos pontos do software
- Podem surgir problemas oriundos de testes mal planejados ou executados
- Pequenas alterações no software exigem muitas alterações nos testes

Solução

- Utilizar Verificação Formal para diminuir casos de testes
- Pares de definição-uso de objetos extraídos durante a verificação
- Java Pathfinder: ferramenta já existente que verifica se um software satisfaz ou não certas propriedades
- Implementar nova funcionalidade que extraia informações acerca do comportamento do software durante a execução do Java Pathfinder
- Pontos cobertos pelo verificador não precisam ser novamente testados

Projeto Maior

- Este projeto é parte de um projeto maior da área de Engenharia de Software



Java PathfinderX

- Verificador de modelo desenvolvido pela Nasa para garantir certas propriedades do programa alvo
- Recebe um programa Java e um arquivo contendo todos os pares de definição-uso dos objetos do programa e devolve arquivos com todos pares cobertos e não cobertos dos objetos do programa alvo, além de uma listagem com todas instruções executadas
- Análise dos resultados permite diminuir o número de casos de testes necessários para validar o software

Exemplo Java Pathfinder

Programa Java

```
public class DeadlockException {
    static Lock lock1;
    static Lock lock2;
    static int state;

    public static void main(String[] args) {
        lock1 = new Lock();
        lock2 = new Lock();
        Process1 p1 = new Process1();
        Process2 p2 = new Process2();
        p1.start();
        p2.start();
    }

    public static void geraException() throws Exception {
        int valor = (int) (Math.random()*10);
        if(valor%2 == 0)
            throw new Exception();
    }
}

class Process1 extends Thread {
    public void run() {
        DeadlockException.state++;
        synchronized (DeadlockException.lock2) {
            synchronized (DeadlockException.lock1) {
                DeadlockException.state++;
            }
        }
    }
}
```

Java Pathfinder

Saída na Tela

```
----- transition #14 thread: 2
gov.nasa.ipf.jvm.choice.ThreadChoiceFromSet (Thread-0,>Thread-1)
  DeadlockException.java:58      : synchronized (DeadlockException.lock1) {
----- transition #15 thread: 2
gov.nasa.ipf.jvm.choice.ThreadChoiceFromSet (Thread-0,>Thread-1)
  DeadlockException.java:58      : synchronized (DeadlockException.lock1) {
  DeadlockException.java:59      : synchronized (DeadlockException.lock2) {
===== snapshot #1
thread index=1,name=Thread-
0,status=BLOCKED,this=Process1@223,priority=5,lockCount=0
owned locks:Lock@219
blocked on: Lock@218
call stack:
  at Process1.run(DeadlockException.java:39)

thread index=2,name=Thread-
1,status=BLOCKED,this=Process2@248,priority=5,lockCount=0
owned locks:Lock@218
blocked on: Lock@219
call stack:
  at Process2.run(DeadlockException.java:59)

===== results
error #1: gov.nasa.ipf.jvm.NotDeadlockedProperty "deadlock encountered: thread
inde..."

===== search finished: 06/11/07
17:18
```

Exemplo Java PathfinderX

Programa Java + Arquivo OcongraX

```
public class DeadlockException {
    static Lock lock1;
    static Lock lock2;
    static int state;

    public static void main(String[] args) {
        lock1 = new Lock();
        lock2 = new Lock();
        Process1 p1 = new Process1();
        Process2 p2 = new Process2();
        p1.start();
        p2.start();
    }

    public static void geraException() throws Exception {
        int valor = (int) (Math.random()*10);
        if(valor%2 == 0)
            throw new Exception();
    }
}

class Process1 extends Thread {
    public void run() {
        DeadlockException.state++;
        synchronized (DeadlockException.lock2) {
            synchronized (DeadlockException.lock1) {
                DeadlockException.state++;
            }
        }
    }
}

class Process2 extends Thread {
    public void run() {
        DeadlockException.state++;
        synchronized (DeadlockException.lock1) {
            synchronized (DeadlockException.lock2) {
                DeadlockException.state++;
            }
        }
    }
}
```

Saída na Tela + Arquivos XML + Arquivos Threads

```
----- transition #14 thread: 2
gov.nasa.jpfi.jvm.choice.ThreadChoiceFromSet (Thread-0,Thread-1)
DeadlockException.java:58 : synchronized (DeadlockException.lock1) {
----- transition #15 thread: 2
gov.nasa.jpfi.jvm.choice.ThreadChoiceFromSet (Thread-0,Thread-1)
DeadlockException.java:58 : synchronized (DeadlockException.lock1) {
DeadlockException.java:59 : synchronized (DeadlockException.lock2) {

----- snapshot #1
Thread index=1, name=Thread-0, status=BLOCKED, this=Process1@223, priority=5, lockCount=0
owned locks: lock0219
blocked on: lock0218
call stack:
    at Process1.run (DeadlockException.java:39)

Thread index=2, name=Thread-1, status=BLOCKED, this=Process2@248, priority=5, lockCount=0
owned locks: lock0218
blocked on: lock0219
call stack:
    at Process2.run (DeadlockException.java:59)

----- results
error #1: gov.nasa.jpfi.jvm.NotDeadlockedProperty "deadlock encountered: thread
inde..."
----- search finished: 06/11/07
#7:16
```

+

Java PathfinderX

```
<?xml version="1.0" encoding="iso-8859-1"?>
<elements>
  <object>
    <classname>name1</classname>
    <name>teste</name>
    <pair>
      <def>23</def>
      <use>24</use>
    </pair>
    <pair>
      <def>25</def>
      <use>37</use>
    </pair>
    <pair>
      <def>23</def>
      <use>100</use>
    </pair>
    <def>1</def>
    <use>2</use>
  </object>
  <object>
    <classname>name2</classname>
    <name>teste2</name>
    <pair>
      <def>24</def>
      <use>23</use>
    </pair>
    <pair>
      <def>26</def>
      <use>25</use>
    </pair>
  </object>
</elements>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<elements>
  <object>
    <classname>name1</classname>
    <name>teste</name>
    <pair>
      <def>5</def>
      <use>11</use>
    </pair>
  </object>
  <object>
    <classname>name2</classname>
    <name>teste2</name>
    <pair>
      <def>14</def>
      <use>23</use>
    </pair>
  </object>
  <exception>
    <classname>name1</classname>
    <name>teste3</name>
    <pair>
      <def>18</def>
      <use>36</use>
    </pair>
  </exception>
</elements>
```

+

```
15 DeadlockException.state++;
16 synchronized (DeadlockException.lock2) {
17   synchronized (DeadlockException.lock1) {
18     DeadlockException.state++;
19     synchronized (DeadlockException.lock1) {
20       synchronized (DeadlockException.lock2) {
21         DeadlockException.state++;
22         synchronized (DeadlockException.lock1) {
23           synchronized (DeadlockException.lock2) {
24             DeadlockException.state++;
25             synchronized (DeadlockException.lock1) {
26               synchronized (DeadlockException.lock2) {
27                 DeadlockException.state++;
28                 synchronized (DeadlockException.lock1) {
29                   synchronized (DeadlockException.lock2) {
30                     DeadlockException.state++;
31                     synchronized (DeadlockException.lock1) {
32                       synchronized (DeadlockException.lock2) {
33                         DeadlockException.state++;
34                         synchronized (DeadlockException.lock1) {
35                           synchronized (DeadlockException.lock2) {
36                             DeadlockException.state++;
37                             synchronized (DeadlockException.lock1) {
38                               synchronized (DeadlockException.lock2) {
39                                 DeadlockException.state++;
40                                 synchronized (DeadlockException.lock1) {
41                                   synchronized (DeadlockException.lock2) {
42                                     DeadlockException.state++;
43                                     synchronized (DeadlockException.lock1) {
44                                       synchronized (DeadlockException.lock2) {
45                                         DeadlockException.state++;
46                                         synchronized (DeadlockException.lock1) {
47                                           synchronized (DeadlockException.lock2) {
48                                             DeadlockException.state++;
49                                             synchronized (DeadlockException.lock1) {
50                                               synchronized (DeadlockException.lock2) {
51                                                 DeadlockException.state++;
52                                                 synchronized (DeadlockException.lock1) {
53                                                   synchronized (DeadlockException.lock2) {
54                                                     DeadlockException.state++;
55                                                     synchronized (DeadlockException.lock1) {
56                                                       synchronized (DeadlockException.lock2) {
57                                                         DeadlockException.state++;
58                                                         synchronized (DeadlockException.lock1) {
59                                                           synchronized (DeadlockException.lock2) {
60                                                             DeadlockException.state++;
61                                                             synchronized (DeadlockException.lock1) {
62                                                               synchronized (DeadlockException.lock2) {
63                                                                 DeadlockException.state++;
64                                                                 synchronized (DeadlockException.lock1) {
65                                                                  ...
66                                                                ...
67                                                              ...
68                                                            ...
69                                                          ...
70                                                        ...
71                                                      ...
72                                                    ...
73                                                  ...
74                                                  ...
75                                                  ...
76                                                  ...
77                                                  ...
78                                                  ...
79                                                  ...
80                                                  ...
81                                                  ...
82                                                  ...
83                                                  ...
84                                                  ...
85                                                  ...
86                                                  ...
87                                                  ...
88                                                  ...
89                                                  ...
90                                                  ...
91                                                  ...
92                                                  ...
93                                                  ...
94                                                  ...
95                                                  ...
96                                                  ...
97                                                  ...
98                                                  ...
99                                                  ...
100                                                 ...
101                                               ...
102                                             ...
103                                           ...
104                                         ...
105                                       ...
106                                     ...
107                                   ...
108                                 ...
109                               ...
110                             ...
111                           ...
112                         ...
113                       ...
114                     ...
115                   ...
116                 ...
117               ...
118             ...
119           ...
120         ...
121       ...
122     ...
123   ...
124 }
125 DeadlockException.state++;
```


Exemplo Java PathfinderX

Arquivo XML dos pares cobertos

```
<?xml version="1.0" encoding="iso-8859-1"?>
<elements>
  <object>
    <classname>name1</classname>
    <name>teste</name>
    <pair>
      <def>5</def>
      <use>11</use>
    </pair>
  </object>
  <object>
    <classname>name2</classname>
    <name>teste2</name>
    <pair>
      <def>14</def>
      <use>23</use>
    </pair>
  </object>
  <exception>
    <classname>name1</classname>
    <name>teste3</name>
    <pair>
      <def>18</def>
      <use>36</use>
    </pair>
  </exception>
</elements>
```

Arquivo XML dos pares não cobertos

```
<?xml version="1.0" encoding="iso-8859-1"?>
<elements>
  <object>
    <classname>name1</classname>
    <name>teste</name>
    <pair>
      <def>78</def>
      <use>93</use>
    </pair>
  </object>
  <exception>
    <classname>name1</classname>
    <name>teste3</name>
    <pair>
      <def>21</def>
      <use>42</use>
    </pair>
  </exception>
</elements>
```

Disciplinas Úteis

- MAC0332 – Engenharia de Software
- MAC0340 – Laboratório de Engenharia de Software
- MAC0211 – Laboratório de Programação I
- MAC0242 – Laboratório de Programação II
- MAC0442 – Programação Orientada a Objetos
- MAC0338 – Análise de Algoritmos

Agradecimentos

- Ana Cristina Vieira de Melo
- Kléber da Silva Xavier
- Simone Hanazumi