Parallel Many-Objective Search for Unit Tests

Verena Bader
University of Passau
Germany

José Campos*
University of Washington
USA

Gordon Fraser
University of Passau
Germany

12th IEEE International Conference on Software Testing (ICST)
Xi’an, China
// Given
BankAccount bankAccount0 = new BankAccount();
bankAccount0.deposit(10);
// Given
BankAccount bankAccount0 = new BankAccount();
bankAccount0.deposit(10);

// When
boolean boolean0 = bankAccount0.withdraw(25);
// Given
BankAccount bankAccount0 = new BankAccount();
bankAccount0.deposit(10);

// When
boolean boolean0 = bankAccount0.withdraw(25);

// Then
assertThat(boolean0, false);
Meta-heuristic search algorithms for unit test generation
Initial Population

MOSA
@Test
public void test0() {
    int var0 = 10;
    YearMonthDay var1 = new YearMonthDay(var0);
    DateTime var2 = new TimeOfDay();
    DateTime var3 = var1.toDateTime(var2);
    DateTime var4 = var3.minus(var0);
    DateTime var5 = var4.plusSeconds(var0);
}

@Test
public void test1() {
    DateTime var0 = new DateTime("11-09-2017");
    DateTime var1 = new DateTime("25-12-2017");
    int var2 = DateTime.sub(var0, var1);
}

Initial Population

MOSA

Evaluation
<table>
<thead>
<tr>
<th>test</th>
<th>branch b1</th>
<th>branch b2</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>test 0</td>
<td>0.35</td>
<td>0.10</td>
<td>Rank 0</td>
</tr>
<tr>
<td>test 1</td>
<td>0.15</td>
<td>0.35</td>
<td>Rank 0</td>
</tr>
<tr>
<td>test 2</td>
<td>0.80</td>
<td>0.90</td>
<td>Rank 2</td>
</tr>
<tr>
<td>test 3</td>
<td>0.20</td>
<td>0.25</td>
<td>Rank 1</td>
</tr>
</tbody>
</table>
Offspring → Evaluation → MOSA → Selection
@Test
public void test2() {
    int var0 = 10;
    YearMonthDay var1 = new YearMonthDay(var0);
    DateTime var2 = new TimeOfDay();
    DateTime var3 = var1.toDateTime(var2);
    DateTime var4 = var3.minus(var0);
    DateTime var5 = var4.plusSeconds(var0);
    DateTime var3 = new DateTime("11-09-2017");
    DateTime var4 = new DateTime("25-12-2017");
    int var5 = DateTime.sub(var3, var4);
}
@Test
class Test {
    @Test
    public void test0() {
        int var0 = 10;
        YearMonthDay var1 = new YearMonthDay(var0);
        DateTime var2 = new TimeOfDay();
        DateTime var3 = var1.toDate_time(var2);
        DateTime var4 = var3.minus(var0);
        DateTime var5 = var4.plusSeconds(var0);
        DateTime var4 = new DateTime("11-09-2017");
        DateTime var5 = new DateTime("25-12-2017");
    }

    @Test
    public void test3() {
        DateTime var0 = new TimeOfDay();
        DateTime var1 = new TimeOfDay();
        int var2 = DateTime.add(var0, var1);
    }
}
Offspring → MOSA → Evaluation → Selection
Parallel Genetic Algorithm
Offspring → Evaluation → Selection
Parallel Many-Objective Sorting Algorithm
Initial Population

pMOSA

Evaluation

Selection

UNIVERSITY of WASHINGTON
Empirical Evaluation
Tuning

34 Java classes

216 configurations

pMOSA

60 seconds

30 repetitions
# Parameter tuning

<table>
<thead>
<tr>
<th>Tuned values</th>
</tr>
</thead>
<tbody>
<tr>
<td># Clients</td>
</tr>
<tr>
<td>Migration frequency</td>
</tr>
<tr>
<td>Migration rate</td>
</tr>
<tr>
<td>Migration selection</td>
</tr>
<tr>
<td>function</td>
</tr>
</tbody>
</table>
## Parameter tuning

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Tuned values</th>
<th>Best values</th>
</tr>
</thead>
<tbody>
<tr>
<td># Clients</td>
<td>2, 4, 8</td>
<td>2, 4, 8</td>
</tr>
<tr>
<td>Migration frequency</td>
<td>1, 5, 10, 25</td>
<td>25, 5, 10</td>
</tr>
<tr>
<td>Migration rate</td>
<td>1, 5, 10, 15, 20, 25</td>
<td>20, 1, 1</td>
</tr>
<tr>
<td>Migration selection function</td>
<td>random, best, rank</td>
<td>random, rank, random</td>
</tr>
</tbody>
</table>
Tuning

34 Java classes
216 configurations
pMOSA

Sequential vs Parallel

312 Java classes
2 algorithms
MOSA, pMOSA

60 seconds
30 repetitions
RQ1 - Does parallel MOSA improve over sequential MOSA?
Improvements over sequential

![Box plot showing effect size for different numbers of clients](image)

- **Effect size**
  - 0.2
  - 0.3
  - 0.4
  - 0.5
  - 0.6
  - 0.7
  - 0.8
  - 0.9
  - 1.0

- **Number of clients**
  - 2
  - 4
  - 8

The graph illustrates the improvements in effect size over sequential methods with an increasing number of clients.
Improvements over sequential

Significantly higher coverage on 80 classes
Improvements over sequential

2 → Significantly higher coverage on 80 classes

4 → Significantly higher coverage on 133 classes
Improvements over sequential

Significantly higher coverage on 80 classes

Significantly higher coverage on 133 classes

Significantly higher coverage on 142 classes
Improvements over sequential

Significantly higher coverage on 80 classes

Coverage increases from 79% to 84% using parallelisation with 8 clients

Significantly higher coverage on 142 classes
RQ2 - Does migration contribute to better performance?
Migration vs non-migration

The diagram above shows the comparison of effect size between migration and non-migration for different numbers of clients (2, 4, and 8). The x-axis represents the number of clients, while the y-axis represents the effect size. The box plots indicate the distribution of effect sizes for each condition.
Migration has a positive effect on coverage and it is larger the more sub-populations are independently evolved.
RQ3 - Does parallelisation reduce the overall runtime to achieve coverage?
Runtime reduction

Runtime reduction of the system with different numbers of clients (2, 4, and 8) over time (in seconds). The box plots show the distribution of runtime reduction at various time intervals.
Runtime reduction
Runtime reduction

![Box plot showing runtime reduction across different time intervals and client counts. The x-axis represents seconds, and the y-axis represents the runtime ratio. The box plots indicate the distribution of runtime reductions for 2, 4, and 8 clients.](image-url)
Parallelisation can be used to reduce the overall runtime of the search, but cannot replace the sequential effects of evolution.
Parallel vs Sequential

Offspring → Evaluation → Selection

pMOSA

Effect size

2 4 8

# clients

Parallel vs Sequential

Effect size

0.2

0.3

0.4

0.5

0.6

0.7

0.8

0.9

1.0

2 4 8

# clients

pMOSA

Offspring

Evaluation

Selection
Parallel vs Sequential

Migration vs non-migration
Parallel vs Sequential

Migration vs non-migration

Runtime reduction
Parallel vs Sequential

Migration vs non-migration

Runtime reduction

http://www.evosuite.org/