Team Members

Yunlu Liaozheng 001470321

Zhe Liu 001475826

Problem To Solve

We are trying to use genetic algorithm to solve boxing problem.

There are many elements with different volumes and unlimited boxes with fixed capacity. We are trying to figure out the minimal count of boxes needed to load all elements.

Architecture And Implementation Details

- `com.info6205.ga.*` : Contains the general genetic algorithm system.
    - `GeneticAlgorithm` : The main class which controls the logic of evolution.
        - Configurable parameters : `parentChromosomesSurvivePercentage`, `mutatePossibility` and `crossoverPercentage`.
        - Concurrence : It uses `FixedThreadPool(availableProcessors())` to concurrently calculate mutation and crossover.
        - Iteration(Evolution) listener : It supports `preIterationListener` and `postIterationListener` to monitor and control the process of iteration.
        - Loggers : There are two loggers. One is `com.info6205.ga.GeneticAlgorithm` which is used to log main process. Another is `com.info6205.ga.GeneticAlgorithm.executor` which is used to monitor concurrent process.
    - `Chromosome` : An interface needed to be implemented by user. User should implement `mutate`, `crossover` and `clone` methods. And as it is designed to be immutable, all three methods should return new instance(s).
    - `Fitness` : An interface needed to be implemented by user. User should implement `calculate` method which accepts a `Class extends Chromosome` and returns a `Comparable`.
    - `Population` : A class used to manage a set(list) of `Chromosome`. It uses `List.sort` to sort `Chromosomes` by their fitness. The reason why not use `PriorityQueue` is that actually sorting is not a big deal as we just need to sort limited elements. While the real problem is calculating fitness, and sometimes it can be every costly. So we design to concurrently calculate all fitness first, and then use calculated fitness to sort original `Chromosome` list. Although we haven't implement this.
- `BoxVector` : A custom class extends `Chromosome`.

    - Assumption : The capacity of box is fixed as 1. The size of items won't be larger than 1.
    - Encoding : Accept an array, and every element represents its size.
        - `itemSize` : An array given by user to represent the size of each element.
        - `vector` : An array which is randomly initialized, representing which box this item is located in.
    - Mutation : Randomly change only one element from `0` to `items.length - 1`.
    - Crossover : Randomly exchange some elements of two chromosomes based on given `crossoverPercentage`.
    - Fitness : The count of used boxes, if the total volume of elements in one box exceeds its capacity, return `Integer.MAX_VALUE`. As calculating the fitness of one chromosome relies on and only relies on itself. So here we directly calculate it inside the `BoxVector`.

- `BoxFitness` : A custom class extends `Fitness`.

    - Calculation : Just call `BoxVector.getUsedBoxes`.

- `Main` : For test

Test Result

Given items :

```
1.0 / 7, 1.0 / 7, 1.0 / 7, 1.0 / 7, 1.0 / 7, 1.0 / 7,
1.0 / 3, 1.0 / 3, 1.0 / 3, 1.0 / 3, 1.0 / 3, 1.0 / 3,
1.0 / 2, 1.0 / 2, 1.0 / 2, 1.0 / 2, 1.0 / 2, 1.0 / 2
```

The best result should be `6`.

Given the following configuration :

```
populationSize = 500
parentChromosomesSurvivePercentage = 1
mutatePossibility = 0.4
crossoverPercentage = 0.5
maxIterations = 500
```

GeneticAlgorithm Output :

```
17:17:06.499 [main] INFO  Main - The best solution with 1 iterations uses 9 boxes.
17:17:06.509 [main] INFO  Main - The best solution with 2 iterations uses 9 boxes.
17:17:06.515 [main] INFO  Main - The best solution with 3 iterations uses 8 boxes.
17:17:06.518 [main] INFO  Main - The best solution with 4 iterations uses 8 boxes.
17:17:06.524 [main] INFO  Main - The best solution with 5 iterations uses 8 boxes.
17:17:06.528 [main] INFO  Main - The best solution with 6 iterations uses 7 boxes.
17:17:06.531 [main] INFO  Main - The best solution with 7 iterations uses 7 boxes.
```

```
17:17:06.534 [main] INFO  Main - The best solution with 8 iterations uses 7 boxes.
17:17:06.537 [main] INFO  Main - The best solution with 9 iterations uses 7 boxes.
17:17:06.542 [main] INFO  Main - The best solution with 10 iterations uses 7 boxes.
17:17:06.545 [main] INFO  Main - The best solution with 11 iterations uses 7 boxes.
17:17:06.549 [main] INFO  Main - The best solution with 12 iterations uses 7 boxes.
17:17:06.552 [main] INFO  Main - The best solution with 13 iterations uses 7 boxes.
17:17:06.555 [main] INFO  Main - The best solution with 14 iterations uses 7 boxes.
17:17:06.558 [main] INFO  Main - The best solution with 15 iterations uses 7 boxes.
17:17:06.561 [main] INFO  Main - The best solution with 16 iterations uses 7 boxes.
17:17:06.564 [main] INFO  Main - The best solution with 17 iterations uses 7 boxes.
17:17:06.568 [main] INFO  Main - The best solution with 18 iterations uses 6 boxes.
17:17:06.569 [main] DEBUG com.info6205.ga.GeneticAlgorithm - Evolution terminates,
threadpool has been shutdown.
```

How lucky we are!

Concurrence Evidence

Config `com.info6205.ga.GeneticAlgorithm.executor`'s level to `trace`, and part of the result:

```
17:18:46.163 [pool-2-thread-6] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 0
17:18:46.162 [pool-2-thread-5] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent mutation with index: 129
17:18:46.163 [pool-2-thread-5] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 2
17:18:46.163 [pool-2-thread-10] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 3
17:18:46.163 [pool-2-thread-5] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 4
17:18:46.163 [pool-2-thread-10] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 5
17:18:46.163 [pool-2-thread-5] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 6
17:18:46.162 [pool-2-thread-11] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent mutation with index: 101
17:18:46.163 [pool-2-thread-5] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 8
17:18:46.162 [pool-2-thread-9] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent mutation with index: 149
17:18:46.163 [pool-2-thread-5] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 9
17:18:46.163 [pool-2-thread-11] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 10
17:18:46.163 [pool-2-thread-9] TRACE com.info6205.ga.GeneticAlgorithm.executor -
Starting concurrent crossover with index: 11
```