

**2019年春季学期  
计算机学院《软件构造》课程**

**Lab 6实验报告**

|  |  |
| --- | --- |
| 姓名 | 郭子阳 |
| 学号 | 1170300520 |
| 班号 | 1703005 |
| 电子邮件 | [guoziyang0033@gmail.com](mailto:guoziyang0033@gmail.com) |
| 手机号码 | 18800420598 |

**目录**

[1 实验目标概述 1](#_Toc1393927)

[2 实验环境配置 1](#_Toc1393928)

[3 实验过程 1](#_Toc1393929)

[3.1 ADT设计方案 1](#_Toc1393930)

[3.2 Monkey线程的run()的执行流程图 1](#_Toc1393931)

[3.3 至少两种“梯子选择”策略的设计与实现方案 2](#_Toc1393932)

[3.3.1 策略1 2](#_Toc1393933)

[3.3.2 策略2 2](#_Toc1393934)

[3.3.3 策略3（可选） 2](#_Toc1393935)

[3.4 “猴子生成器”MonkeyGenerator 2](#_Toc1393936)

[3.5 如何确保threadsafe？ 2](#_Toc1393937)

[3.6 系统吞吐率和公平性的度量方案 2](#_Toc1393938)

[3.7 输出方案设计 2](#_Toc1393939)

[3.8 猴子过河模拟器v1 2](#_Toc1393940)

[3.8.1 参数如何初始化 2](#_Toc1393941)

[3.8.2 使用Strategy模式为每只猴子随机选择决策策略 2](#_Toc1393942)

[3.9 猴子过河模拟器v2 2](#_Toc1393943)

[3.9.1 对比分析：固定其他参数，选择不同的决策策略 3](#_Toc1393944)

[3.9.2 对比分析：变化某个参数，固定其他参数 3](#_Toc1393945)

[3.9.3 分析：吞吐率是否与各参数/决策策略有相关性？ 3](#_Toc1393946)

[3.9.4 压力测试结果与分析 3](#_Toc1393947)

[4 实验进度记录 3](#_Toc1393948)

[5 实验过程中遇到的困难与解决途径 3](#_Toc1393949)

[6 实验过程中收获的经验、教训、感想 4](#_Toc1393950)

[6.1 实验过程中收获的经验和教训 4](#_Toc1393951)

[6.2 针对以下方面的感受 4](#_Toc1393952)

# 实验目标概述

本次实验训练学生的并行编程的基本能力，特别是Java多线程编程的能力。

根据一个具体需求，开发两个版本的模拟器，仔细选择保证线程安全（threadsafe）的构造策略并在代码中加以实现，通过实际数据模拟，测试程序是否是线程安全的。另外，训练学生如何在threadsafe和性能之间寻求较优的折中，为此计算吞吐率和公平性等性能指标，并做仿真实验。

* Java 多线程编程
* 面向线程安全的 ADT 设计策略选择、文档化
* 模拟仿真实验与对比分析

# 实验环境配置

实验环境：

系统：macOS Mojave 10.14.3，Oracle JDK 1.8.0\_211

ide：Eclipse IDE for Eclipse Committers 2018-12 (4.10.0), IntelliJ IDEA ULTIMATE 2019.1

编辑器：Visual Studio Code 1.31.1

构建工具：Gradle，Travis CI，Drone CI

版本管理：git

代码托管：Github, Gogs

配置过程：

本次实验环境无需特殊配置，使用Lab5的实验环境即可

GitHub Lab6仓库的URL地址

https://github.com/ComputerScienceHIT/Lab6-1170300520

# 实验过程

## ADT设计方案

设计了猴子Monkey类，梯子Ladder类

Monkey类用于具体模拟每一只猴子，具有的属性有id、方向和速度。

Ladder类用于模拟梯子，具有的属性有id和长度。

猴子以数组的形式存储在Ladder中，如果一个Ladder的某个位置为空，那么数组中的该位置为null。

Ladder暴露了一些比较便捷的接口，用于对当前梯子上的状况作出判断，例如该梯子是否被占用，该梯子是否可以添加猴子到左（右）端点，该梯子是否有猴子向左（右）行进，获取梯子上猴子的数量，获取梯子的某个方向上的实际移动速度等。

## Monkey线程的run()的执行流程图

Monkey的过河线程为CrossingBridge类，实现了Runnable接口。

进入while(true)循环。首先将标志暂停的Boolean设置为true，接着获取自身的锁，判断标志暂停的Boolean是否为true，如果是就执行wait()等待，直到主线程调用该对象的方法，将暂停的Boolean设为false，并notify()，线程继续，获取所有梯子的锁，首先判断自己是否在梯子上，如果不在，就首先执行选择梯子的策略，再执行移动，否则只执行移动，最后判断线程停止的标志位是否为true，如果为true，就将自身移出线程Set，并且从while(true)中break。

## 至少两种“梯子选择”策略的设计与实现方案

### 策略1

优先选择没有猴子的梯子，若所有梯子上都有猴子，则优先选择没有与我对向而行的猴子的梯子；若满足该条件的梯子有很多，则随机选择；

### 策略2

优先选择没有猴子的梯子，若所有梯子上都有猴子，则优先选择没有与我对向而行且离我最近的猴子的实际速度最快的梯子。

### 策略3（可选）

优先选择没有猴子的梯子，若所有梯子上都有猴子，则优先选择没有与我对向而行且离我最近的猴子的实际速度与我的速度最接近的梯子。

## “猴子生成器”MonkeyGenerator

MonkeyGenerator类并未使用多线程生成猴子，猴子的生成是在主线程中进行的，在主线程中使用while循环调配统一多线程的时刻，由于所有的猴子线程做完决策的时间远远小于一秒，于是主线程的循环每次sleep一秒后便进行到下一时刻。在循环开头，判断当前时刻是否模间隔时间余0，如果是，就调用MonkeyGenerator的生成猴子的方法生成指定数量的猴子。

在生成猴子的方法中，实例化猴子对象和线程对象，并将其加入main类中的一个List中，并且启动这个线程（线程被启动时会首先暂停，直到主线程唤醒）。

## 如何确保threadsafe？

在主线程中，每一秒会遍历一遍过河线程的List，唤醒所有的线程，每一个线程运行的过程中都需要抢占一个Set<Ladder>对象的锁，于是在同一时刻，只有一个线程在做决策，并对修改其中的数据，于是，整个过程是线程安全的。

## 系统吞吐率和公平性的度量方案

吞吐率使用猴子个数除以全部过河结束时的时间。

在每只猴子成功过河后，会在一个Map中存储当前的时间，在所有的过河结束后，遍历每一对猴子，根据猴子的id判断猴子生成的时间，通过Map 获取猴子过河的时间，即可计算出公平性。

## 输出方案设计

日志

日志系统未使用日志框架，而是直接输出在控制台上，在每一秒结束后，输出当前秒的行动信息，如生成猴子的信息、上梯子的信息、移动信息等。

如某一秒的日志如下：

-1 +1 -1 -1 -1 +24 -1 -1 -1 -1 -1 -1 -1 +26 -1 -1 +43 -1 -1 -1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 +12 -1 +17 -1 +37 -1

+34 -1 +30 -1 -1 -1 +14 -1 -1 +25 -1 -1 +18 -1 -1 -1 -1 -1 -1 -1

-1 -1 -1 -1 +23 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 +38 -1 +50 -1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 +8 -1 +32 -1 -1

-1 -1 -1 -1 -1 -1 -1 +13 -1 -1 +7 -1 -1 +48 -1 -1 +27 -1 -1 -1

+40 -1 +33 -1 +5 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1

+47 +45 -1 +29 -1 +10 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 +4 -1

-1 -1 -1 -1 +3 -1 -1 -1 -1 -1 -1 -1 -1 +36 -1 -1 -1 +41 -1 +28

-1 -1 -1 +42 -1 -1 -1 -1 -1 +39 -1 -1 -1 -1 -1 +31 -1 -1 -1 -1

New Monkey id=51 direction=R->L speed=5 strategy=1

New Monkey id=52 direction=L->R speed=6 strategy=2

New Monkey id=53 direction=L->R speed=2 strategy=0

New Monkey id=54 direction=R->L speed=4 strategy=0

New Monkey id=55 direction=L->R speed=6 strategy=1

New Monkey id=56 direction=R->L speed=6 strategy=2

New Monkey id=57 direction=L->R speed=3 strategy=2

New Monkey id=58 direction=R->L speed=2 strategy=1

New Monkey id=59 direction=L->R speed=1 strategy=1

New Monkey id=60 direction=L->R speed=2 strategy=0

monkey 60 is on ladder 9

Monkey 60 on ladder 9 current grid=0 move to right=2

Monkey 33 on ladder 6 current grid=2 move to right=1

monkey 73 is on ladder 3

Monkey 73 on ladder 3 current grid=19 move to left=0

Monkey 47 on ladder 7 current grid=0 move to right=0

Monkey 7 on ladder 5 current grid=10 move to left=2

Monkey 3 on ladder 8 current grid=4 move to left=3

Monkey 50 on ladder 3 current grid=18 move to left=1

Monkey 30 on ladder 2 current grid=2 move to right=1

Monkey 23 on ladder 3 current grid=4 move to left=4

Monkey 32 on ladder 4 current grid=17 move to left=1

monkey 67 is on ladder 5

Monkey 67 on ladder 5 current grid=19 move to left=2

Monkey 26 on ladder 0 current grid=13 move to left=2

。。。

GUI

GUI用于形象动态地表现猴子过河的过程

过河过程：



过河结束后，数据展示：



## 猴子过河模拟器v1

### 参数如何初始化

参数从配置文件中读取，并存入main类的一个静态的配置对象中。配置文件的格式类似：

n=10

h=20

t=3

N=500

k=25

MV=7

使用NIO的内存映射的方式读取，可在程序的任意位置读取。

### 使用Strategy模式为每只猴子选择决策策略

在每只猴子进行决策时，首先判断这只猴子是否已经在梯子上，如果不在，根据猴子的策略调用不同的策略方法，选择梯子即可。

## 猴子过河模拟器v2

在不同参数设置和不同“梯子选择”模式下的“吞吐率”和“公平性”实验结果及其对比分析。

### 对比分析：固定其他参数，选择不同的决策策略

参数固定为：5个梯子，梯子长20格，每三秒生成25只猴子，共生成500只猴子，猴子的最大速度为7。

仅使用策略1，优先选择没有猴子的梯子，若所有梯子上都有猴子，则优先选择没有与我对向而行的猴子的梯子；若满足该条件的梯子有很多，则随机选择；

测试五次

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 第一次 | 第二次 | 第三次 | 第四次 | 第五次 | 平均 |
| 吞吐率 | 2.304 | 2.294 | 2.294 | 2.326 | 2.273 | 2.298 |
| 公平性 | 0.313 | 0.360 | 0.375 | 0.375 | 0.187 | 0.322 |

仅使用策略2，优先选择没有猴子的梯子，若所有梯子上都有猴子，则优先选择没有与我对向而行且离我最近的猴子的实际速度最快的梯子。

测试五次

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 第一次 | 第二次 | 第三次 | 第四次 | 第五次 | 平均 |
| 吞吐率 | 2.326 | 2.315 | 2.283 | 2.326 | 2.294 | 2.309 |
| 公平性 | 0.356 | 0.364 | 0.235 | 0.316 | 0.252 | 0.305 |

仅使用策略3，优先选择没有猴子的梯子，若所有梯子上都有猴子，则优先选择没有与我对向而行且离我最近的猴子的实际速度与我的速度最接近的梯子。

测试五次

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 第一次 | 第二次 | 第三次 | 第四次 | 第五次 | 平均 |
| 吞吐率 | 2.304 | 2.315 | 2.262 | 2.336 | 2.294 | 2.302 |
| 公平性 | 0.361 | 0.318 | 0.301 | 0.368 | 0.359 | 0.341 |

### 对比分析：变化某个参数，固定其他参数

变化梯子的个数n

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | n=5 | n=10 | n=15 | n=20 |
| 吞吐率 | 2.272 | 4.132 | 5.814 | 6.098 |
| 公平性 | 0.349 | 0.627 | 0.898 | 0.867 |

### 分析：吞吐率是否与各参数/决策策略有相关性？

梯子的个数与吞吐率相关性较大，但是在提升到一定程度后，由于梯子足够使用，再增加梯子的意义不大，导致相关性下降。

决策策略与吞吐率有一定关系，但是影响较小，可能由于剔除了效率较低的策略，而使用了效率较高的策略，使差异不大。

### 压力测试结果与分析

500只猴子，每3秒产生100只，共5个梯子，使用策略2

用时218秒，吞吐率2.294，公平性0.334，吞吐率和公平性没有明显下降。

## 猴子过河模拟器v3

针对教师提供的三个文本文件，分别进行多次模拟，记录模拟结果。

|  |  |  |
| --- | --- | --- |
|  | 吞吐率 | 公平性 |
| Competiton\_1.txt |  |  |
| 第1次模拟 | 0.323 | 1 |
| 第2次模拟 | 0.313 | 1 |
| 第3次模拟 | 0.313 | 1 |
| 第4次模拟 | 0.303 | 1 |
| 第5次模拟 | 0.323 | 1 |
| 第6次模拟 | 0.323 | 1 |
| 第7次模拟 | 0.323 | 1 |
| 第8次模拟 | 0.333 | 1 |
| 第9次模拟 | 0.323 | 1 |
| 第10次模拟 | 0.333 | 1 |
| 平均值 | 0.321 | 1 |
| Competiton\_2.txt |  |  |
| 第1次模拟 | 0.400 | 1 |
| 第2次模拟 | 0.357 | 1 |
| 第3次模拟 | 0.357 | 1 |
| 第4次模拟 | 0.385 | 1 |
| 第5次模拟 | 0.357 | 1 |
| 第6次模拟 | 0.370 | 1 |
| 第7次模拟 | 0.357 | 1 |
| 第8次模拟 | 0.357 | 1 |
| 第9次模拟 | 0.370 | 1 |
| 第10次模拟 | 0.400 | 1 |
| 平均值 | 0.371 | 1 |
| Competiton\_3.txt |  |  |
| 第1次模拟 | 0.454 | 1 |
| 第2次模拟 | 0.476 | 1 |
| 第3次模拟 | 0.476 | 1 |
| 第4次模拟 | 0.435 | 1 |
| 第5次模拟 | 0.435 | 1 |
| 第6次模拟 | 0.435 | 1 |
| 第7次模拟 | 0.456 | 1 |
| 第8次模拟 | 0.456 | 1 |
| 第9次模拟 | 0.435 | 1 |
| 第10次模拟 | 0.456 | 1 |
| 平均值 | 0.451 | 1 |

# 实验进度记录

请使用表格方式记录你的进度情况，以超过半小时的连续编程时间为一行。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 计划任务 | 实际完成情况 |
| 2019.5.27 | 16:00-20:00 | 完成配置文件读取 | 按时完成 |
| 2019.5.30 | 18:00-20:30 | 完成GUI演示 | 按时完成 |
| 2019.6.1 | 20:00-22:00 | 修正策略，完成v1 | 按时完成 |
| 2019.6.3 | 19:00-23:30 | 完成v2与v3 | 按时完成 |

# 实验过程中遇到的困难与解决途径

|  |  |
| --- | --- |
| 遇到的难点 | 解决途径 |
| 多线程暂停和继续不熟悉 | 在线学习，查看课件 |

# 实验过程中收获的经验、教训、感想

## 实验过程中收获的经验和教训

学会了并行编程，多线程程序的编写，以及编写程序中对线程安全的考虑。

## 针对以下方面的感受

多线程程序编写时需要考虑线程安全问题，但是在性能上较强。

应当采用一些线程安全的对象来进行程序的编写，或者采取一些措施。

本实验工作量较小，难度适中，时间较长。