2.2071257696525555

**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 | 闫锋 |
| 学号 | 1170300614 |
| 班号 | 1703006 |
| 电子邮件 | [1502833622@qq.com](mailto:1502833622@qq.com) |
| 手机号码 | 18800410681 |

**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

|  |  |
| --- | --- |
|  | 配置 |
| URL | <https://github.com/ComputerScienceHIT/Lab6-1170300614> |
| 系统 | Win10 |
| JDK版本 | 1.8.0\_16 |
| JAVA\_PATH | C:\Program Files (x86)\Common Files\Oracle\Java\javapath |

# 实验过程

请仔细对照实验手册，针对三个问题中的每一项任务，在下面各节中记录你的实验过程、阐述你的设计思路和问题求解思路，可辅之以示意图或关键源代码加以说明（但千万不要把你的源代码全部粘贴过来！）。

## ADT设计方案

设计了哪些ADT、各自的作用、属性、方法；

给出每个ADT的specification；

设计的ADT有Board类，Ladder类，Monkey类，Mylog类，BarChart类，Simulator类和Strategy类。

1.Board类：

属性：

Monkey表示该格子上的猴子，state表示该格子是否被占用。

方法：

setMonkey(Monkey monkey)方法可以向该格梯子放置猴子。

getState()方法可以获取当前梯子格里是否有猴子

2.Ladder类：

属性：list内存储的是该梯子包含的所有格子；State表示该梯子上是否已经有猴子；direction表示当前该梯子上的猴子的运动情况；ID表示该梯子的ID，slowv表示当前该梯子上最后一只猴子的运动速度。

3.Monkey类：

属性：ID是猴子唯一的ID，direction存储的是猴子的运动方向，v是猴子的速度，hpms是猴子的实际运动速度，set内存的是所有猴子共享的梯子集合，beigin存储的是猴子出生的时间，list存储的是所有猴子共享的列表，存储猴子到达终点的顺序。

方法：getSet()方法可以获取内部的梯子集合，getID()方法可以获取猴子的ID，getHpms()方法可以获得猴子的实际运动速度，setHpms()方法可以设置猴子的实际运动速度，getDirection()方法可以获得猴子的运动方向，getV()方法可以获得猴子出生的初始速度，getBegin()可以获得猴子出生的时间，run()方法是猴子过河的过程。

4.Mylog类：

属性：**logger**

作用：在日志中输出模拟过河整个过程的各步骤信息，以及本次模拟的公平性和吞吐率。清晰显示出整个过河过程、本次模拟的各参数设置情况、 总体性能情况。

5.Simulator类：

属性：n是梯子的数目，h是梯子的长度，MV是猴子的最大速度，N是猴子的总数目，t和k满足每隔t秒生成k个猴子（最后一次不包括在内）。

方法：readFile()方法可以从文件读取数据，初始化该类的属性，Initial()初始化所有梯子，MonkeyGenerator()方法可以生成猴子并且运行线程，计算吞吐率和公平性。

6.Strategy类：

作用：提供不同的可以供猴子选择的梯子的策略。

方法：execute(Monkey monkey)根据不同的策略为当前线程的猴子选择合适的梯子。

## Monkey线程的run()的执行流程图

## 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，则随机选择一个梯子，该梯子上面的猴子的运动方向都与当前猴子相同。若所有梯子上的猴子运动方向都与当前猴子运动方向相反，则原地等待。

### 策略2

优先选择整体推进速度最快的梯子（没有与我对向而行的猴 子、其上的猴子数量最少、梯子上离我距离最近的猴子的真实行进速 度最快）； ⚫

### 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，则在岸 边等待，直到某个梯子空闲出来

## “猴子生成器”MonkeyGenerator

仿真时读文件初始化部分参数，然后根据这些参数来生成猴子每隔𝑡秒钟同时产生𝑘个 Monkey 对象（例如：第 0 秒生成𝑘个 Monkey 对象，第𝑡秒又同时产生 𝑘个 Monkey 对象，第 2𝑡秒…），并为各只猴子生成以下属性： ⚫ 名字 ID（int）：按照产生的时间次序进行自然数编号，同一时刻 同时生成的猴子的 ID 应有不同的编号 ⚫ 方向 direction（String）：值随机指定，左岸到右岸(“L->R”)， 或者从右岸到左岸(“R->L”) ⚫ 速度𝑣：正整数，取值范围为[1,𝑀𝑉]，𝑀𝑉为最大可能的速度。

## 系统吞吐率和公平性的度量方案

计算吞吐率和公平性：计算并输出本次仿真的吞吐率和公平性。 ⚫ “吞吐率”是指：假如𝑁只猴子过河的总耗时为𝑇秒，那么每只猴子的 平均耗时为𝑋 = 𝑇 𝑁 秒，则吞吐率𝑇ℎ = 𝑁 𝑇 表征每秒钟可过河的猴子数目。

⚫ “公平性”是指：如果 Monkey 对象 A 比 B 出生得早，那么 A 应该不 晚于 B 抵达对岸，则为“公平”；若 A 比 B 晚到对岸，则为“不公 平”。设 A 和 B 的产生时间分别为𝑌𝑎和𝑌𝑏，抵达对岸的时间分别为𝑍𝑎

和𝑍𝑏，那么公平性𝐹(𝐴,𝐵) = {

1, 𝑖𝑓 (𝑌𝑏 − 𝑌𝑎) ∗ (𝑍𝑏 − 𝑍𝑎) ≥ 0 −1, 𝑜𝑡ℎ𝑒𝑟𝑤𝑖𝑠𝑒 。对𝑁 只猴子两两计算其之间的公平性并综合到一起，得到本次模拟的整体

公平性𝐹 =

∑ 𝐹(𝐴,𝐵) (𝐴,𝐵)∈Θ C𝑁 2 ,Θ = {(𝐴,𝐵)|𝐴 ≠ 𝐵,(𝐵,𝐴) ∉ Θ}，其取值范围

为[−1,1]。

## 输出方案设计

日志

(1)利用日志记录参数设置，比如n,t,k,N,MV等待。在日志最开始打印出来，方便后面分析使用。

(2)利用日志来输出猴子的动态。每隔一秒更新一次猴子的位置，猴子的当前状态分三种情况：①猴子出生，以及猴子出生时的初始速度和运动方向②在岸边等待，离出生已经q毫秒。③正在第i架梯子的第j个踏板上，自左向右(自右向左)前进，离出生已经q毫秒，同时要打印该猴子的初始速度和当前的实际速度。④已经从左向右或者从右向左抵达对岸，耗时一共q毫秒。

Gui

## 猴子过河模拟器v1

### 参数如何初始化

梯子的数目n,长度h，猴子的总数目N，以及猴子的最大速度MV和每次间隔生产猴子的时间t和每次生产的数目k，都是从文件读入的。在Simulator类的readFile()方法中，通过正则匹配获取相应的数值，并且赋值给相应的变量。在仿真的最开始完成该部分参数的初始化。

### 使用Strategy模式为每只猴子选择决策策略

实现Strategy接口，然后在Strategy1，Strategy2和Strategy3中实现该接口中的方法，根据不同的策略选择合适的梯子并返回给Monkey类中的run()方法中的调用。

在生成猴子之后，运行线程初始的时候，生成一个随机数，随机数可以是1,2或者3，分别对应着策略1，策略2和策略3。实现了为每只猴子随机选择策略。

## 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

|  |  |  |  |
| --- | --- | --- | --- |
| （吞吐率,公平性） | 策略1 | 策略2 | 策略3 |
| 实验  (n=5,h=20,t=1,N=20,k=2,MV=5) | 0.6896551724137931  0.5789473684210527 | 0.425531914893617  0.37894736842105264 | 0.625  0.5684210526315789 |

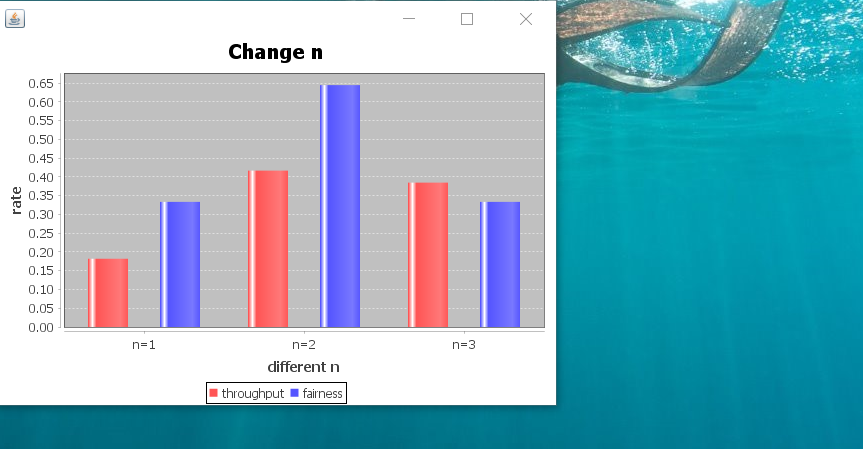
策略3最好，策略1次之，策略2最差。特别是在对于梯子的数目与猴子数目的比值比较小的时候，吞吐率下降很明显。

### 对比分析：变化某个参数，固定其他参数

使用最佳策略3

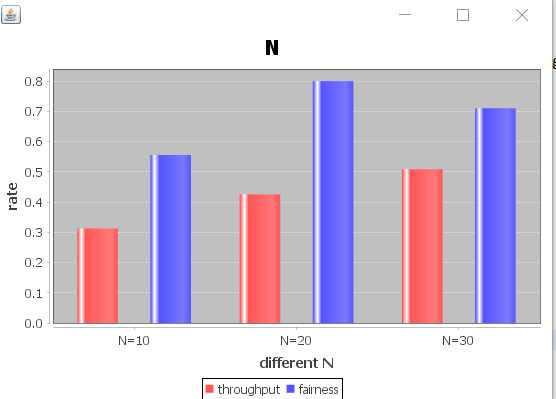
1. 改变n的取值

|  |  |  |
| --- | --- | --- |
|  | 吞吐率 | 公平性 |
| n=1 | 0.18181818181818182 | 0.3333333333333333 |
| n=2 | 0.4166666666666667 | 0.6444444444444445 |
| n=3 | 0.38461538461538464 | 0.3333333333333333 |



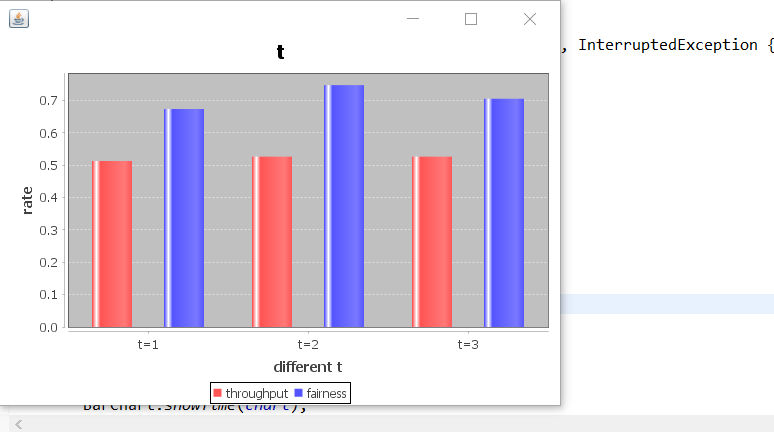
1. 改变N

|  |  |  |
| --- | --- | --- |
|  | 吞吐率 | 公平性 |
| N=10 | 0.3125 | 0.5555555555555556 |
| N=20 | 0.42553191489361 | 0.8 |
| N=30 | 0.5084745762711864 | 0.7103448275862069 |



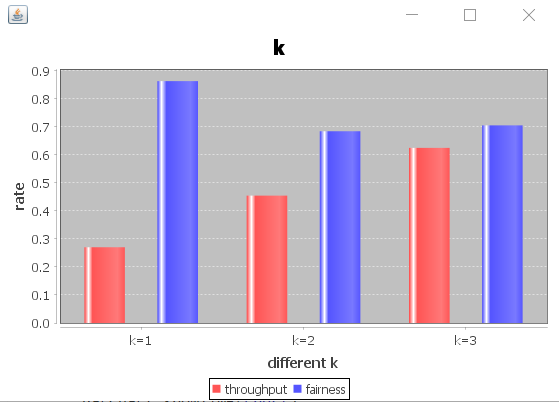
3,改变t

|  |  |  |
| --- | --- | --- |
|  | 吞吐率 | 公平性 |
| t=1 | 0.5128205128205128 | 0.6736842105263158 |
| t=2 | 0.5233157894736842 | 0.7152631578947368 |
| t=3 | 0.5263157894736842 | 0.7052631578947368 |



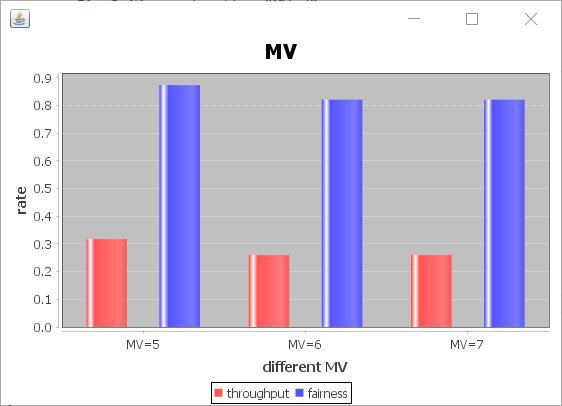
4改变k

|  |  |  |
| --- | --- | --- |
|  | 吞吐率 | 公平性 |
| k=1 | 0.2702702702702703 | 0.821 |
| k=2 | 0.45454545454545453 | 0.6842105263157895 |
| k=3 | 0.625 | 0.7052631578947368 |



5.改变MV

|  |  |  |
| --- | --- | --- |
|  | 吞吐率 | 公平性 |
| MV=5 | 0.31746031746031744 | 0.8736842105263158 |
| MV=6 | 0.2597402597402597 | 0.8210526315789474 |
| MV=7 | 0.2597402597402597 | 0.8210526315789474 |



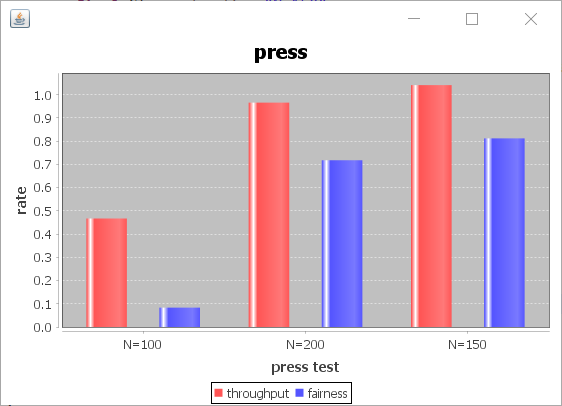
### 分析：吞吐率是否与各参数/决策策略有相关性？

由测试实验可以看到，吞吐率确实与一些参数和策略是有关的。在第一个实验中我们看到，策略1和策略3的吞吐率在不同参数下的表现总是要好于同等情况下的策略2的表现的。同时可以看到，其他参数都不变时，k增加，那么每隔相同时间产生的猴子的数目增加，吞吐率慢慢上升。其他参数都不变时，t增加，那么产生猴子的频率下降，吞吐率也慢慢下降。并且可以看到在一定范围内，N(猴子总数目)的增加也会引发吞吐率的上升。其他参数不变时，n增加，即梯子的数目增加，那么吞吐率也在上升，很明显，因为单位时间可以通过的猴子更多了

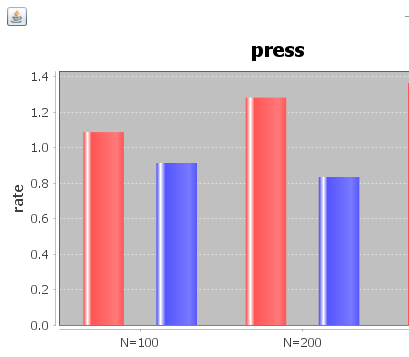
### 压力测试结果与分析

压力测试 1：设计一种参数配置，使得产生的猴子数量非常多、非常密集， 而梯子数量有限。观察此时你的程序的吞吐率和公平性表现如何。

|  |  |  |
| --- | --- | --- |
|  | 吞吐率 | 公平性 |
| n=1,h=20,t=2,N=100,k=3,MV=5 | 0.4672897196261682 | 0.08323232323232323 |
| n=1,h=20,t=2,N=150,k=3,MV=5 | 0.966183574879227 | 0.7181909547738693 |
| n=1,h=20,t=2,N=200,k=3,MV=5 | 1.0416666666666667 | 0.8120805369127517 |



压力测试 2：设计一种参数配置，使得各猴子的速度差异非常大。观察此时 你的程序的吞吐率和公平性表现如何。



## 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 | 2.2071257696525555 | 0.3870457 |
| 第2次模拟 | 2.2227064167362243 | 0.36051282 |
| … |  |  |
| 第10次模拟 | 2.1850662397337683 | 0.38535118 |
| 平均值 | 2.231666211713031 | 0.3619398 |
| Competiton\_2.txt |  |  |
| 第1次模拟 | 4.553789248257551 | 0.5597355 |
| 第2次模拟 | 4.367025421288747 | 0.54232466 |
| … |  |  |
| 第10次模拟 | 4.616163006284134 | 0.55192786 |
| 平均值 | 4.611662901526618 | 0.53592783 |
| Competiton\_3.txt |  |  |
| 第1次模拟 | 0.9863720799047939 | 0.28828284 |
| 第2次模拟 | 1.0165673414515752 | 0.2979798 |
| … |  |  |
| 第10次模拟 | 1.002214391204066 | 0.29313132 |
| 平均值 | 1.016115603634804 | 0.29353535 |

# 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。

每次结束编程时，请向该表格中增加一行。不要事后胡乱填写。

不要嫌烦，该表格可帮助你汇总你在每个任务上付出的时间和精力，发现自己不擅长的任务，后续有意识的弥补。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 计划任务 | 实际完成情况 |
| 6.1 | Free time | 完成ADT | 基本完成 |
| 6.3 | Free day | 实现仿真 | 实现 |
| 6.5 | Free day | 结束实验 | 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 |
| 处理thread -safe | 上网学习 |
|  |  |
|  |  |

# 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

## 针对以下方面的感受

1. 多线程程序比单线程程序复杂在哪里？你是否能体验到多线程程序在性能方面的改善？

答：多线程程序需要比单线程程序多考虑竞争条件的发生，需要保证数据线程安全。但是多线程程序在性能方面要比单线程好，并行比串行效率更高。

1. 你采用了什么设计决策来保证threadsafe？如何做到在threadsafe和性能之间很好的折中？

答：大多数地方用同步的方式来保证thread-safe，部分用的是线程安全的ADT来配合。为了使得thread-safe和性能之间有很好的折中，尽量避免了不必要的同步，使同步块在保证安全的底线下尽可能小。

1. 你在完成本实验过程中是否遇到过线程不安全的情况？你是如何改进的？

答：在本次实验中遇到过线程不安全的情况，策略2时猴子上了梯子的同时就让该线程获得锁，然后设置梯子状态。问

1. 关于本实验的工作量、难度、deadline。
2. 到此为止你对《软件构造》课程的意见和建议。