

**2018年春季学期  
计算机学院大二软件构造课程**

**Lab 6实验报告**

|  |  |
| --- | --- |
| 姓名 | 郭潇 |
| 学号 | 1160300521 |
| 班号 | 03005 |
| 电子邮件 | Wqsnzszw@163.com |
| 手机号码 | 18523735573 |

**目录**

[1 实验目标概述 1](#_Toc517019206)

[2 实验环境配置 1](#_Toc517019207)

[3 实验过程 1](#_Toc517019208)

[3.1 ADT设计方案 1](#_Toc517019209)

[3.2 Monkey线程的run()的执行流程图 1](#_Toc517019210)

[3.3 至少两种“梯子选择”策略的设计与实现方案 1](#_Toc517019211)

[3.3.1 策略1 1](#_Toc517019212)

[3.3.2 策略2 1](#_Toc517019213)

[3.3.3 策略3（可选） 1](#_Toc517019214)

[3.4 “猴子生成器”MonkeyGenerator 1](#_Toc517019215)

[3.5 如何确保threadsafe？ 1](#_Toc517019216)

[3.6 系统吞吐率和公平性的度量方案 1](#_Toc517019217)

[3.7 输出方案设计 1](#_Toc517019218)

[3.8 猴子过河模拟器v1 1](#_Toc517019219)

[3.8.1 参数如何初始化 1](#_Toc517019220)

[3.8.2 使用Strategy模式为每只猴子随机选择决策策略 1](#_Toc517019221)

[3.9 猴子过河模拟器v2 1](#_Toc517019222)

[3.9.1 对比分析：固定其他参数，选择不同的决策策略 1](#_Toc517019223)

[3.9.2 对比分析：变化某个参数，固定其他参数 1](#_Toc517019224)

[3.9.3 分析：吞吐率是否与各参数/决策策略有相关性？ 1](#_Toc517019225)

[3.9.4 压力测试结果与分析 1](#_Toc517019226)

[4 实验进度记录 1](#_Toc517019227)

[5 实验过程中遇到的困难与解决途径 1](#_Toc517019228)

[6 实验过程中收获的经验、教训、感想 1](#_Toc517019229)

# 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 和运行性能之间寻求较优的折中，为此计 算吞吐率等性能指标，并做仿真实验。

1.Java多线程编程

2．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

3.基本的GUI编程

# 实验环境配置

简要陈述你配置本次实验所需环境的过程，必要时可以给出屏幕截图。

特别是要记录配置过程中遇到的问题和困难，以及如何解决的。

在这里给出你的GitHub Lab6仓库的URL地址（Lab6-学号）。

https://github.com/ComputerScienceHIT/Lab6-1160300521

# 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；

Monkey：monkey用于创建每个猴子对象，属性有ID、方向、状态（等待、过桥中、已过桥）、速度和记录猴子所在的梯子。

方法有：猴子所在梯子的设置和获取；ID的获取；方向的获取；速度的获取；tostring；在梯子上的前进；梯子的选择（采用strategy策略）；

Ladder：ladder用于创建每个梯子对象，属性有ID、长度、方向和记录猴子位置的map。

方法有：获取当前未被占用的前n个阶梯数；map的获取；ID的获取；猴子在梯子上的前进；猴子走上梯子；

给出每个ADT的specification；

Monkey：

1. RI为满足lable不为null且入度和出度均大于0
2. AF为符合上述要求的所有点
3. Safety from Rep Expouser为所有变量为private；所有的变量通过get方法观察；remove方法只返回true或false；add方法返回不变量或数字0

Ladder：

1.RI为满足vertex不含有空白字符也不为空白字符

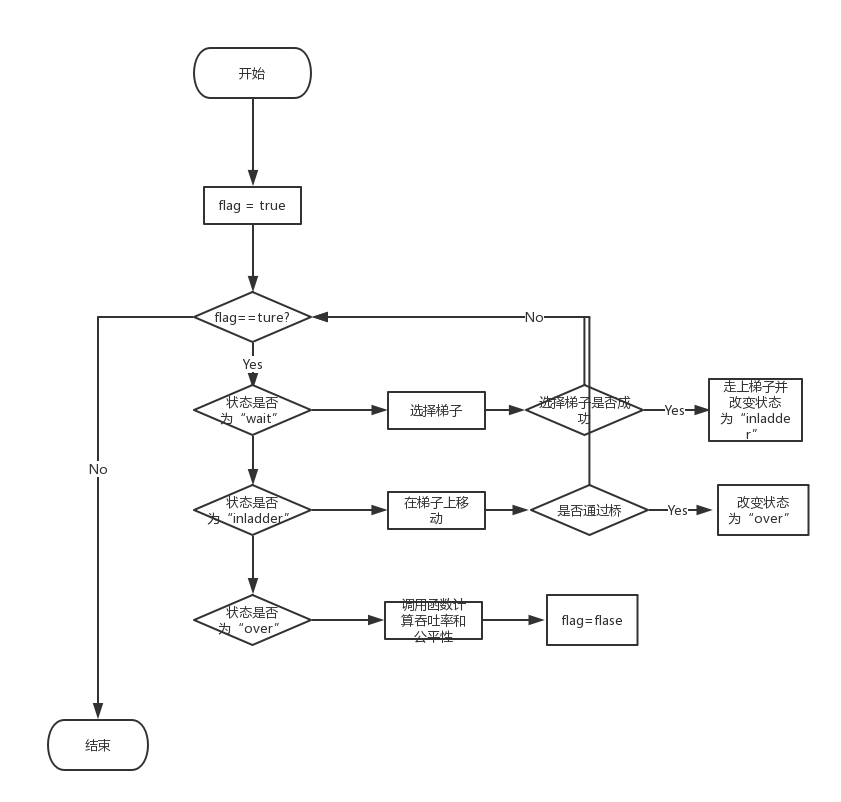
2.AF为符合上述要求的所有顶点

3.Safety from Rep Expouser为所有变量为private；所有类型是mutable的返回值，均创建新的变量返回防止泄露

（可选）以类图形式给出多个类之间的关系。

## Monkey线程的run()的执行流程图

这里无需考虑具体采用的梯子选择策略。



## 至少两种“梯子选择”策略的设计与实现方案

### 策略1

遍历所有的梯子，选择方向相同的且猴子数量最少的桥。

### 策略2

遍历所有梯子，选择前n个阶梯未被占用数量最多的梯子，确保推进速度最快。

### 策略3（可选）

遍历所有梯子，选择猴子数量为0的梯子。

## “猴子生成器”MonkeyGenerator

MonkeyGenerator方法有：读取文件中参数生成参数范围内属性的猴子和桥；以及产生猴子线程的产生；按规定的间隔时间产生相应的猴子线程。

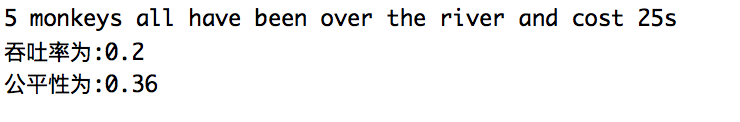
先读取参数文件得到相应参数，然后利用参数生成规定范围内的猴子和桥，然后根据猴子的数量，按照参数时间和数量，为每只猴子生成线程。

## 如何确保threadsafe？

因为每个猴子之间是独立的，包含自己的属性和所选择的梯子；所有梯子之间的数据是共享的，其上的每个猴子移动后都需要将其数据进行更新，所有对Ladder对象加入锁，防止一个梯子上的两只猴子对其同时进行更新。

## 系统吞吐率和公平性的度量方案

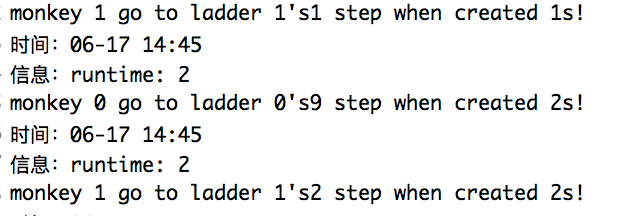
加入一个fair类，用于计算吞吐率和公平性。具体实现为先记录猴子总数，在每次猴子过河完成后，加入到fiar类的一个hashmap中，key为猴子编号，value为过河时间。若加入后hashmap的size等于猴子数，那么可知所有猴子过河完成，此时按猴子编号遍历所有值，若大编号对应的值减去小编号对应的值为正数，表示后生成的猴子后过河，F(a，b) = 1；若为负数，表示先生成的猴子后过河，F(a，b) = -1，将所有猴子遍历完后，计算公平性并输出。通过将猴子的数量除以总的过河时间得到吞吐率并输出。在fair类中也加入了锁，防止过河的猴子同时向hashmap中加入数据。输出结果如下：

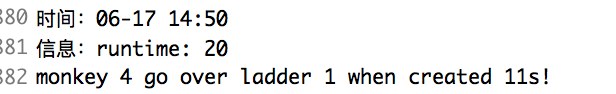


## 输出方案设计

利用上一个实验的log，对于每个线程的猴子，每隔一秒写入一次状态，包括：猴子在左岸、右岸等待；猴子在编号为n的ladder上的位置以及猴子过完河。记录每个状态时，同时记录当前过河用时和猴子的据生成的时间，具体如下：







## 猴子过河模拟器v1

### 参数如何初始化

利用**MonkeyGenerator，读取相应的配置文件，利用字符串的replace和spilt方法获取参数，参数的格式如下：**

****

**然后依次调用build方法，利用随机数和输入的范围，随机生成猴子的方向、速度和策略，然后调用creatmonkeysbytime方法进行生成猴子的操作。**

### 使用Strategy模式为每只猴子随机选择决策策略

先写一个名为Choosebehavior的接口，再在monkey类中加入一个Choosebehavior对象且并未实例化，在随机生成猴子的过程中，利用生成的随机数的值，为每个猴子添加Choosebehavior示例。在线程中，猴子选择桥时，调用其operate方法，operate方法进一步调用Choosebehavior的方法，实现各个猴子有不同的选择策略。

## 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。

与v1相比，v2需要在生成猴子时，按照所给的参数，为猴子设定选择策略。

### 对比分析：固定其他参数，选择不同的决策策略

在n = 3;h = 20;t = 3;N = 50;k = 2;MV = 5;的参数配置下进行比较：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.6024096385542169 | 0.4716 |
| 2 | 0.5555555555555556 | 0.3228 |
| 3 | 0.3184713375796178 | 0.2212 |

### 对比分析：变化某个参数，固定其他参数

在n = 3;h = 20;t = 3;N = 50;k = 2;MV = 5;的参数配置下进行调整记录相关数据：

增大n为5：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.5208333333333334 | 0.4404 |
| 2 | 0.5555555555555556 | 0.4628 |
| 3 | 0.4672897196261682 | 0.39 |

减小n为2：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.5494505494505495 | 0.4748 |
| 2 | 0.4716981132075472 | 0.358 |
| 3 | 0.20920502092050208 | 0.1628 |

增大N为80：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.5882352941176471 | 0.46875 |
| 2 | 0.6106870229007634 | 0.4696875 |
| 3 | 0.29739776951672864 | 0.2690625 |

减小N为20：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.4444444444444444 | 0.405 |
| 2 | 0.4166666666666667 | 0.395 |
| 3 | 0.3333333333333333 | 0.345 |

增大t为5：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.36496350364963503 | 0.4876 |
| 2 | 0.3424657534246575 | 0.4812 |
| 3 | 0.2717391304347826 | 0.4012 |

减小t为2：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.704225352112676 | 0.4516 |
| 2 | 0.704225352112676 | 0.45 |
| 3 | 0.26737967914438504 | 0.2668 |

增大k为3：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.7575757575757576 | 0.4364 |
| 2 | 0.6329113924050633 | 0.2444 |
| 3 | 0.32679738562091504 | 0.1916 |

减小k为1：

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.29239766081871343 | 0.4844 |
| 2 | 0.3225806451612903 | 0.4884 |
| 3 | 0.3105590062111801 | 0.4764 |

### 分析：吞吐率是否与各参数/决策策略有相关性？

对比上述实验所获得的数据，分析可知：

1. 对吞吐率影响较大的参数是t和k，t=2与t=5相比，吞吐率足足提高了一倍；k=3与k=1相比，吞吐率提高了比一倍还多。
2. 猴子的数量N增大能提高吞吐率，但是提高与t、k相比，并不明显。
3. 桥的数量n对吞吐率的影响很小。
4. 决策1、2的吞吐率在不同参数配置下的差距并不明显，而策略3因为只是选择没有猴子的梯子，过于简单，所以效率很低。

### 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，而梯子数量压力测试

选取的参数配置为：n = 2;h = 20;t = 1;N = 200;k = 5;MV = 5

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 1.0869565217391304 | 0.16005 |
| 2 | 1.0101010101010102 | 0.1315 |
| 3 | 0.2290950744558992 | 0.05465 |

观察可知：对于策略1、2，增大了N、k,减小t都会增大吞吐率，所以此情况下吞吐率会很大，而公平性在猴子多，桥少的情况下，会有很大猴子在两岸等待，而选择桥的顺序是随机的，所以一旦后生成的猴子先上桥，那么会有很大的优势先于其他猴子过桥，所以公平性变得很小。

对于策略3，因为猴子会选择没有被猴子的桥，所以相当于一次只能等一只猴子过完桥以后下一只猴子才能上桥，所以吞吐率会减小，且公平性因为等待的猴子过多，减小也就更明显。

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。

选取的参数配置为：n = 4;h = 20;t = 3;N = 80;k = 2;MV = 15（且速度只能为3的倍数）;

|  |  |  |
| --- | --- | --- |
| 策略 | 吞吐率 | 公平性 |
| 1 | 0.6451612903225806 | 0.494375 |
| 2 | 0.64 | 0.4975 |
| 3 | 0.6451612903225806 | 0.4978125 |

观察可知：因为猴子之间速度的差异很大，若先生成的猴子速度很快的话，那么能够较快的过河，保证两岸等待的猴子数量更少，那么后生成的猴子先于先生成的猴子上桥的概率会变变小，那么公平性能够得到很好的保证。对于策略3，在这种速度差距大的情况下，因为速度快的猴子多的话会使得桥上的猴子数量变小，那么特定时刻下能够选到没有猴子的桥的概率也会增大，所以策略3能够得到与1、2差不多的性能。

# 实验进度记录

请尽可能详细的记录你的进度情况。

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 时间段 | 计划任务 | 实际完成情况 |
| 2018-06-06 | 18：00-21：30 | 完成猴子过河的单线程版本 | 完成 |
| 2018-06-07 | 16：00-17：30 | 将单线程版本改写为简单多线程版本 | 未解决竞争问题 |
| 2018-06-07 | 18：00-21：00 | 解决线程间的竞争问题并编写读取文件和MonkeyGenerator | 完成 |
| 2018-06-08 | 19：00-21：00 | 加入猴子的决策和日志记录部分 | 完成 |
| 2018-06-09 | 18：30-21：00 | 加入决策的strategy模式，吞吐率和公平性计算部分， | 完成 |
| 2018-06-10 | 14：00-17：00 | 完成v2部分 | 完成 |
| 2018-06-17 | 15：00-18：00 | 完成实验报告 | 完成 |
|  |  |  |  |
|  |  |  |  |

# 实验过程中遇到的困难与解决途径

1. 不了解线程产生和消除竞争相关内容，导致死锁和无法终止的情况发生

解决：多次试验发现死锁是因为猴子选择桥时的状态修改出错，无法终止是因为没有正确统计过河猴子的数量。于是复习ppt和查阅相关资料，稍微改进便完成了要求。

2.计算吞吐率和公平性时，因为各个线程间相互独立，所以难以统计各猴子过河的时间

解决：在计算吞吐率和公平性时，新写一个类，记录每只猴子过河的时间与编号关系，再所有猴子过完河后进行计算。并在该类中加入锁以消除竞争。同时发现，若使用List记录猴子过河时间，若后产生的猴子比先产生的猴子过完河，那么会先向list中加入编号大的书库，这样会造成越界，所以采用map记录过河时间。

# 实验过程中收获的经验、教训、感想

本节除了总结你在实验过程中收获的经验和教训，也可就以下方面谈谈你的感受（非必须）：

本次实验是我第一次接触多线程编程，主要知道了如何产生进程和消除简单的竞争。意识到多线程程序相比单线程程序编程上的更困难，同时也意思到了一些多线程的优势。就实验提供的问题来说多线程更能解决问题。对于多线程相比单线程的优势，我认为有以下几点：

1. 多线程能充分运用计算机的性能，在模拟猴子过河问题中有多个进程同时进行，相比单个计算能够节省大量时间。
2. 多线程能够简化程序，猴子过河问题中若考虑单线程执行，那么肯定会产生许多复杂的关系，而使用多线程能够在消除竞争的前提下使得程序更简单。
3. 更方便与优化程序，在猴子过河问题中，改进猴子选择桥的问题变回产生很大的吞吐率差异，所以多线程程序虽然更简单，但其性能可能会被一个简单因素所影响。