用于列车售票的可线性化并发数据结构

2022 年 9 月 21 日

给定 Ticket 类：

```
class Ticket {
    long tid;
    String passenger;
    int route;
    int coach;
    int seat;
    int departure;
    int arrival;
}
```

其中，tid 是车票编号，passenger 是乘客名字，route 是列车车次，coach 是车厢号，seat 是座位号，departure 是出发站编号，arrival 是到达站编号。

给定 TicketingSystem 接口：

```
public interface TicketingSystem {
    Ticket buyTicket(String passenger,int route,
                     int departure,int arrival);
    int inquiry(int route,int departure,int arrival);
    boolean refundTicket(Ticket ticket);
}
```

其中，

- buyTicket 是购票方法，即乘客 passenger 购买 route 车次从 departure 站到 arrival 站的车票 1 张。若购票成功，返回有效的 Ticket 对象；若失败（即无余票），返回空对象（即 return null）。
- refundTicket 是退票方法，对有效的 Ticket 对象返回 true，对错误或无效的 Ticket 对象返回 false。

- inquriy 是查询余票方法，即查询 route 车次从 departure 站到 arrival 站的余票数。

每位学生使用 Java 语言设计并完成一个用于列车售票的可线性化并发数据结构：TicketingDS 类，该类实现 TicketingSystem 接口，同时提供

```
TicketingDS(routenum, coachnum, seatnum, stationnum, threadnum);
```

构造函数。其中，routenum 是车次总数（缺省为 5），coachnum 是每次列车的车厢数目（缺省为 8），seatnum 是每节车厢的座位数（缺省为 100），stationnum 是每个车次经停站的数量（缺省为 10，含始发站和终点站），threadnum 是并发购票的线程数（缺省为 16）。

为简单起见，假设每个车次的 coachnum、seatnum 和 stationnum 都相同。车票涉及的各项参数均从 1 开始计数，例如车厢从 1 到 8 编号，车站从 1 到 10 编号等。

每位学生需编写多线程测试程序，在 main 方法中用下述语句创建 TicketingDS 类的一个实例。

```
final TicketingDS tds = new
TicketingDS(routenum, coachnum, seatnum, stationnum, threadnum);
```

系统中同时存在 threadnum（缺省为 16）个线程，每个线程是一个票务代理，按照 60% 查询余票、30% 购票和 10% 退票的比率反复调用 TicketingDS 类的三种方法若干次（缺省为总共 10000 次）。按照线程数为 4、8、16、32、64 的情况分别给出每种方法调用的平均执行时间，同时计算系统的总吞吐率（单位时间内完成的方法调用总数）。

1 正确性要求

- 每张车票都有一个唯一的编号 tid，不能重复。
- 每一个 tid 的车票只能出售一次。退票后，原车票的 tid 作废。
- 每个区段有余票时，系统必须满足该区段的购票请求。
- 车票不能超卖，系统不能卖无座车票。
- 买票、退票和查询余票方法都需满足可线性化要求。

2 作业评分标准

作业评分包括三部分，基本分（50%），性能分 (50%) 和奖励分。

1. 首先保证并发数据结构功能正确。如果发现实现有错误，只能按照完成情况给基本分。

2. 对于所有正确实现的并发数据结构，用统一的多线程基准程序在同一测试环境下测试系统的性能（包括延迟和吞吐率），并按照并发数据结构的性能测试结果进行加权排序，从高到低依次给出性能分。

3. **大作业需本人独立完成，不得抄袭他人代码。如果发现抄袭，成绩为零分。**

3 作业清单

大作业按照 myproject 目录打包提交，程序编码为 UTF-8 格式（要求程序必须在 Linux 系统上能够正常编译和运行）。

大作业提交 package 的名字为 ticketingsystem，所有 Java 程序放在 ticketingsystem 目录中，history.sh 文件放在 ticketingsystem 目录的上层目录 myproject 中。如果程序有多重目录，那么将主 Java 程序放在 ticketingsystem 目录中。至少包含 5 个文件 (见附件 myproject.tgz)：

1. TicketingSystem.java 是规范文件，不能更改。

2. GenerateHistory.java 是 history 生成程序，用于正确性验证，不能更改。

3. history.sh 是 history 生成脚本，用于正确性验证，不能更改。

4. TicketingDS.java 是并发数据结构的实现。

5. Test.java 实现多线程性能测试。

3.1 TicketingSystem.java

```
package ticketingsystem;

class Ticket{
        long tid;
        String passenger;
        int route;
        int coach;
        int seat;
```

```
        int departure;
        int arrival;
}

public interface TicketingSystem {
    Ticket buyTicket(String passenger,int route,
        int departure,int arrival);
    int inquiry(int route,int departure,int arrival);
    boolean refundTicket(Ticket ticket);
}
```

3.2 TicketingDS.java

```
package ticketingsystem;

public class TicketingDS implements TicketingSystem {
    //ToDo
}
```

3.3 Test.java

```
package ticketingsystem;

public static void main(String[] args) {

final TicketingDS tds = new
TicketingDS(routenum, coachnum, seatnum, stationnum, threadnum);
    //ToDo
}
```

3.4 history.sh

```
#!/bin/sh
javac -encoding UTF-8 -cp . ticketingsystem/GenerateHistory.java
java -cp . ticketingsystem/GenerateHistory
```

大作业提交前需编译测试通过，要求能正确执行 history.sh 脚本文件生成符合要求的 history（不得更改 history.sh 所在的目录位置），history 的格式见图 1。

```
203465398 203467180 2 RemainTicket 5 2 3 5
203398237 203416399 0 TicketSoldOut 1 1 5
203440102 203443003 1 RemainTicket 0 1 1 4
203544530 203546967 0 RemainTicket 5 3 4 5
203510164 203511162 2 TicketRefund 870 passenger307 2 1 4 5 2
203600713 203605960 0 TicketBought 932 passenger369 3 2 4 5 2
203486636 203487657 3 TicketRefund 868 passenger373 1 1 4 5 4
203649129 203650451 0 TicketRefund 834 passenger925 2 2 2 3 3
203620931 203632948 2 TicketSoldOut 1 2 4
203566474 203568190 1 RemainTicket 9 2 4 5
203710491 203712195 2 TicketRefund 832 passenger651 1 1 4 5 3
203689715 203692399 0 RemainTicket 3 3 3 5
203763643 203764698 2 TicketRefund 820 passenger312 2 3 1 4 1
203737539 203738865 1 TicketRefund 924 passenger978 1 2 2 3 2
203822821 203826718 2 TicketBought 933 passenger229 1 1 4 5 3
203796668 203798018 0 TicketRefund 879 passenger730 1 2 4 5 1
203799081 203800725 3 RemainTicket 10 1 1 2
203914911 203916606 0 TicketRefund 880 passenger930 1 2 3 5 4
203873441 203874969 2 RemainTicket 0 1 2 5
203974057 203978499 0 TicketBought 934 passenger909 2 1 3 5 2
203848243 203851209 1 RemainTicket 0 1 2 5
203992362 203996391 2 TicketBought 935 passenger690 2 1 1 3 1
203937960 203939058 3 TicketRefund 930 passenger158 2 2 3 5 1
204067144 204070810 2 TicketBought 936 passenger952 3 3 2 5 4
```

图 1: sample of a history

3.5 性能评测报告

大作业同时提交性能评测报告：阐述并发数据结构和多线程测试程序的设计思路，分析系统的正确性和性能，解释所实现的每个方法如何满足可线性化、是否 deadlock-free、starvation-free、lock-free 或 wait-free。