

# LINUX OPERATING SYSTEM

YANG

LINUX操作系统（双语）





双语课→课件内容中英混排



## |Lecture 7

# Synchronization

# 本讲内容

 并发

 竞争

 协作

 异步产生的错误

 同步



# 并发

# 并发进程/线程




- 💡 在内存中同时存在的若干个进程/线程，由操作系统的调度程序采用适当的策略将他(们)调度至CPU(s)上运行，同时维护他们的状态队列。
- 💡 多个并发进程/线程从宏观上是同时在运行；
- 💡 从微观上看，他们的运行过程是走走停停；
- 💡 并发的进程/线程之间是交替执行（Interleaving）。
- 💡 注：此后课中不再刻意区分并发进程和并发线程，他们都可以理解为运行的实体和调度的单位。

# 并发进程之间的关系

## 独立关系

-  并发进程分别在自己的变量集合上运行
-  例如：chrome进程和music进程

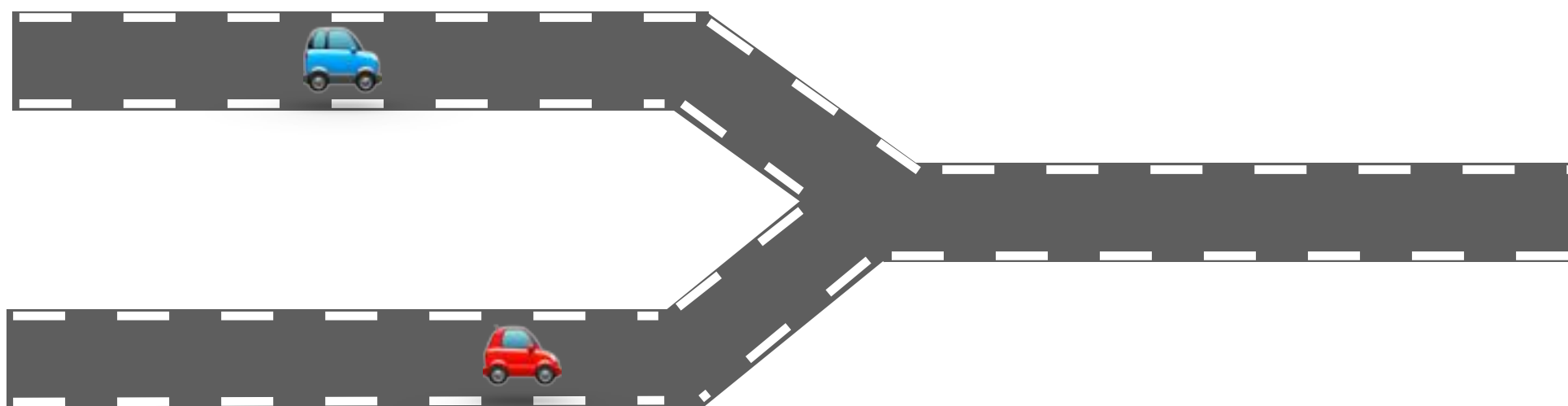
## 交互关系

-  并发进程执行过程中需要共享或是交换数据
-  例如：银行交易服务器上的receiver进程和handler进程
-  交互的并发进程之间又存在着**竞争**和**协作**的关系

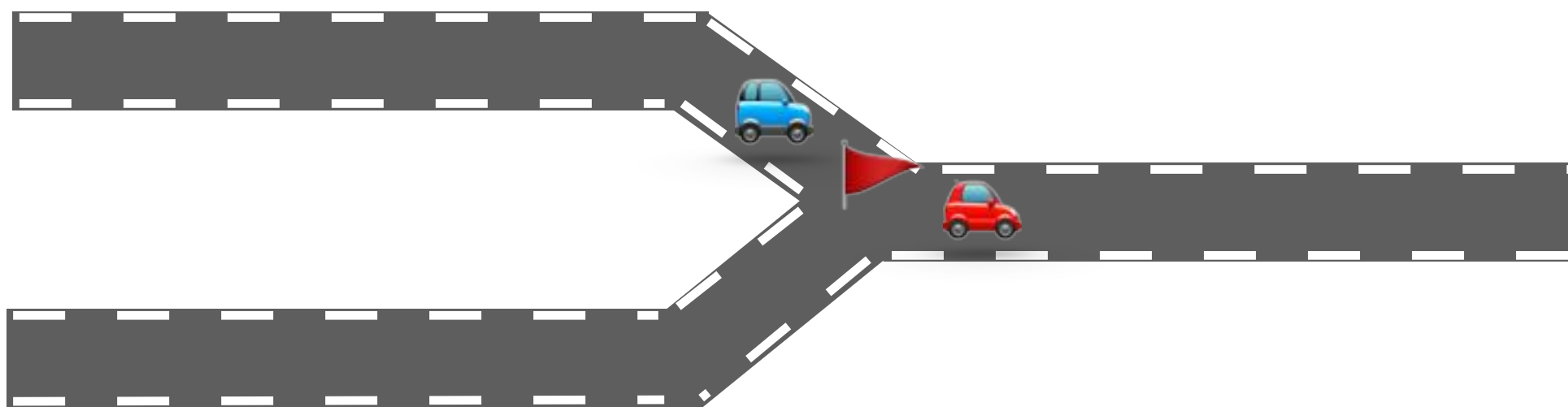
# 竞争



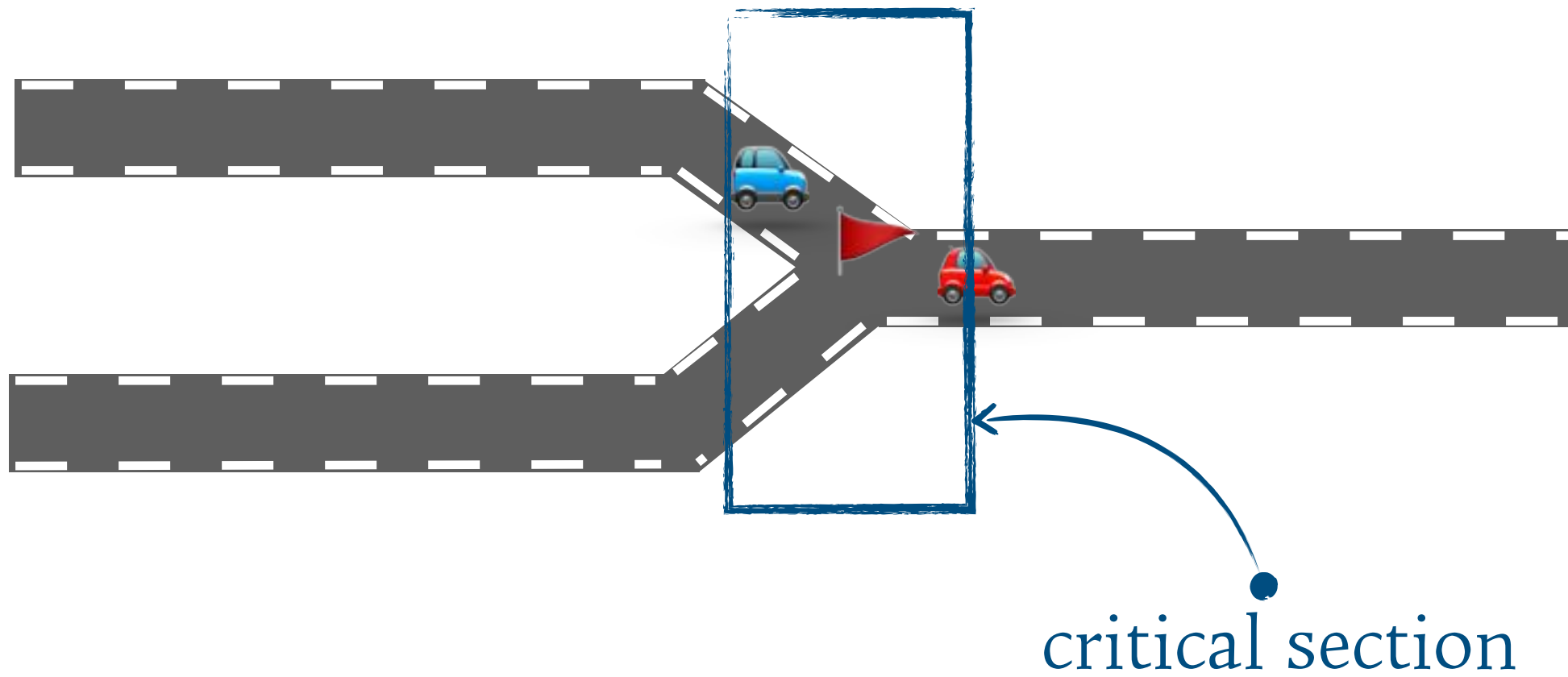
# 竞争 ( RACE )



# 竞争 ( RACE )

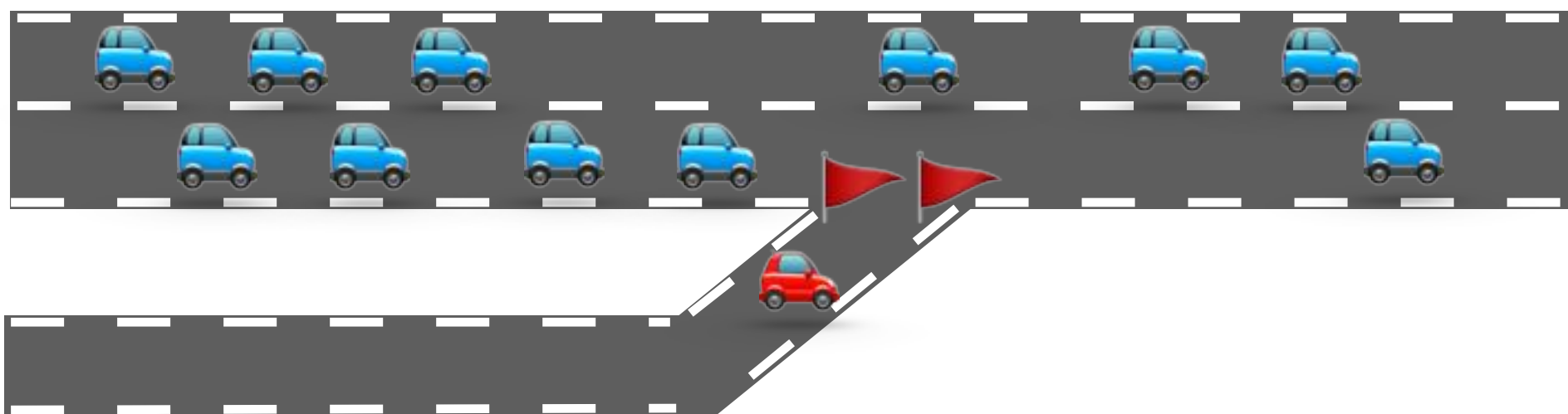


# 竞争 ( RACE )



# 协作

# 协作 ( COOPERATION )



主干道每过2辆车才允许匝道通过1辆

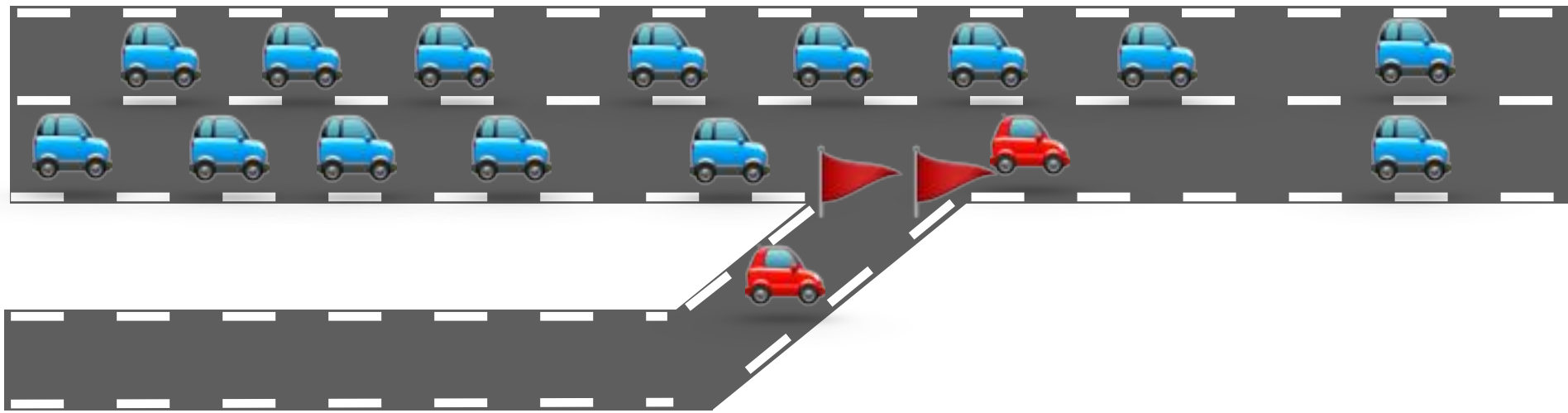
# 协作 ( COOPERATION )



# 协作 ( COOPERATION )



# 协作 ( COOPERATION )





# 异步产生的错误

# 异步

💡 Asynchronous means RANDOM!

💡 会引发**竞争条件**（Race Condition）：一种这样的情况：  
多个进程并发操作同一个数据导致执行结果依赖于特定的  
进程执行顺序。

$T_i$ 是订票终端， $x = 2$  代表剩余票数，为所有终端共享，终端逻辑代码如下：

$T_1$  :

.....

$T=x;$

if( $T \geq 1$ ) {

$x = T-1;$

}

.....

$T_2$  :

.....

$T=x;$

if( $T \geq 1$ ) {

$x = T-1;$

}

.....

剩余票数最终结果不确定！可能是1，也可能是0.

# 同步

# 同步

 **Process Synchronization** means a mechanism to maintain the consistency of data shared in cooperative processes.

 Synchronization Tool Kits

 Mutex lock

 Semaphore

# 下期预告

 下次直播时间：3月2日 上午9:30

 课程内容

 Lecture 8 & Practice 3 Mutex Lock



# |Lecture 7

The End

TAKE A BREAK

