《通信网理论与技术基础》教材配套实验

爱尔兰等待制系统

实验指导书

1 实验目标

利用 OMNeT++软件，构建爱尔兰等待制系统的仿真，并对平均服务的顾客数、平均在队列中等待的顾客数和平均等待时间进行数值分析验证。

2 实验背景

爱尔兰等待制系统是指输入为一个泊松流，有 s 个服务员，服务时间满足指数分布，系统容量为无限大的一类排队系统。当所有的服务员都繁忙时，新到的顾客进入等待队列中，队列的排队规则是先入先出（FIFO）。只要有服务员空闲，等待队列中队首的顾客接受服务。

3 实验方法提示

3.1 理论值计算

如下的理论值计算均是在 a<s 的条件下讨论的，其中 a 代表呼叫量、s 代表服务员的个数。

(1) 平均被服务的顾客数

平均被服务的顾客数与呼叫量的值相同。呼叫量的理论值为：

$$a = \frac{\lambda}{\mu} \qquad \text{（式 1）}$$

其中λ代表顾客的到达率，μ代表单个服务员的服务率。

(2) 平均在队列中等待的顾客数

由状态转移图分析得，

$$\begin{cases} p_0 = \dfrac{1}{\sum_{k=0}^{s-1}\dfrac{a^k}{k!} + \dfrac{a^s}{s!}\cdot\dfrac{1}{1-a/s}} \\ \mathrm{C(s,a)} = \dfrac{\mathrm{a^s}}{\mathrm{s!}}\dfrac{\mathrm{p_0}}{1-\mathrm{a/s}} \\ \rho = \dfrac{\lambda}{s\mu} \end{cases}$$

其中，λ代表顾客的到达率，μ代表单个服务员的服务率，s 代表中继线的条数。因此平均在队列中等待的顾客数理论值为：

$$\frac{\rho}{1-\rho}C(s,a) \qquad （式 2）$$

(3) 等待时间ω

等待时间表示顾客从进入系统直至去服务员处接受服务的时间，等待时间的均值 E[ω]为：

$$C(s,a)\cdot\frac{1}{s\mu-\lambda} \qquad （式 3）$$

3.2 仿真

(1) 仿真图

在 OMNeT++软件中，编写简单模块 Source、Exchange、Server 和 Sink，建立仿真。其中 Source 模块负责产生到达率为 lambda 的泊松流；Exchange 模块在所有服务员都繁忙时将顾客纳入排队队列中、在有空闲的服务员时将顾客从队列中取出送至对应的服务员处接受服务；Server 模块负责模拟服务员，按照服务率 mu 服务顾客；Sink 模块负责模拟顾客的离开。

在 Exchange 模块中构建三个统计量，分别负责统计平均被服务的顾客数、平均在队列中等待的顾客数和顾客平均等待时间。

仿真图如下：

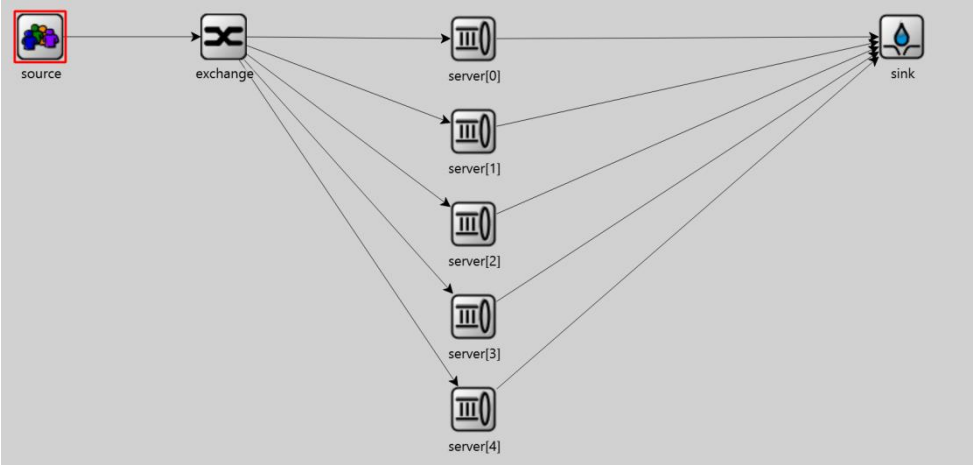

图 1　“爱尔兰等待制系统”的仿真图

(2) 参数设置

在表 1 中，参数“lambda”代表 Source 模块中泊松流的到达率；参数“mu”代表 Server 模块服务员服务顾客的服务率；参数“numServers”代表服务员的个数，该参数不在简单模块中，而是在用来定义复合模块的“MMs.ned”文件中。具体在“MMs.ned”文件中使用该变量的方法参见“实验 4 爱尔兰损失制系统”。

表 1　“爱尔兰等待制系统”的仿真参数设置

网络模块	参数名
Source	lambda
Server	mu
——	numServers

(3) 代码提示——监听信号

创建顾客的方法，参见“实验 3”代码提示部分的“cMessage 类”；在 Qtenv 仿真中自定义中继线条数的方法，参见“实验 4”的代码提示部分；利用 cQueue 容器类存储呼叫的方法，参见“实验 6”的代码提示部分。

在这个实验中，我们需要在 Exchange 简单模块中监听 Server 模块的忙闲状态。当监听到 Server 从繁忙变为空闲时，Exchange 简单模块需要立即将排队队列中队首的顾客送至空闲的 Server 中。

在 OMNeT++中，我们可以利用订阅信号来实现这个功能。

- 在 Server 模块注册并发射（emit()函数）该信号

构建信号名称为 idle，信号 ID 为 idleSignal 的信号。该信号值的类型为布尔类型，当繁忙时，利用 emit()函数发送“false”；当空闲时，利用 emit()函数发送“true”。

参考代码如下：

```
emit(idleSignal, true);
emit(idleSignal, false);
```

- 在 Exchange 模块订阅信号

在模块中引入“Server.h”文件，并在“void Exchange::initialize()”函数中利用如下代码订阅信号。

```
getParentModule()->subscribe("idle", this);
```

- 在 Exchange 模块监听该信号

在模块中重写 cIListener 类中的函数，实现监听。

```
// 监听中继线的忙闲状态,"1"代表空闲，"0"代表繁忙
void Exchange::receiveSignal(cComponent *source, simsignal_t signalID, bool value, cObject *details) {
    // 获取监听对象的索引，从 0 开始
    int serverIndex = check_and_cast<cModule*>(source)->getIndex();
    // 更新对应 server 的空闲状态，serverStatusMap 为一个 map。"键"代表中继线的索引，"值"代表中继线的忙闲状态
    serverStatusMap[serverIndex] = value;
    //启动一个新的活动并将事件上下文更改为当前模块
    Enter_Method_Silent();
}
void Exchange::receiveSignal(cComponent *source, simsignal_t signalID, intval_t i, cObject *details){};
void Exchange::receiveSignal(cComponent *source, simsignal_t signalID, uintval_t i, cObject *details){};
void Exchange::receiveSignal(cComponent *source, simsignal_t signalID, double d, cObject *details){};
void Exchange::receiveSignal(cComponent *source, simsignal_t signalID, const SimTime& t, cObject *details) {};
void Exchange::receiveSignal(cComponent *source, simsignal_t signalID, const char *s, cObject *details){};
void Exchange::receiveSignal(cComponent *source, simsignal_t signalID, cObject *obj, cObject *details) {};
```

4 实验步骤

1）编写 Source、Exchange、Server 和 Sink 四个简单模块；

2）利用编写的四个简单模块构建网络拓扑结构；

3）设置参数的值，进行仿真；

4）仿真结束后，利用基于信号的统计量绘图，将仿真值与理论值进行比较。

5 提交要求

学生需要提交一份实验报告，并将实验报告、源代码、数据打包成压缩包上交。其中实验报告内容包括但不限于：

1）研究背景与问题描述

2）总体思路与方案论证

3）结果分析与结论

4）课程学习收获与体会建议