Queueing in a Bank

Summary

Queueing system is a common model in our life. This time, our goal is to build a mathematical model of a 150-people queueing system which meets the manager's requirement of an average queue length less than 2 people and an average waiting time less than 2 minutes to improve customer satisfactory.

To analyze this queueing system, we put forward two models which can verify each other, they are: the mathematical model and the simulation model. In the mathematical model we use the knowledge of probability theory, hoping to figure out a recursion formula about the probability of the state of queue at a certain time. In order to reach this goal, we perfectly described the state of queue at a certain time by defining several variables. Using this model, we can figure out the two important parameters, which are the average length of queue and the average waiting time.

Then we put forward a simulation model. In the given situation, we put forward an algorithm which can simulate and measure the whole process of how the 150 customers come and leave. Using Monte Carlo method, we simulated the process for many times and then got the average length of queue as well as the average waiting time.

After comparison, the average queue length and waiting time got from the two models coincided with each other perfectly, the average waiting time is about 4.9 minutes while the average queue length is about 2.4 people. Since the simulation model calculated faster, we decided to take it as the only used analyzing model in the following discussions.

Results above showed that the current situation couldn't meet the manager's requirement, so we put forward several solutions from different perspectives to meet the manager's requirement and improve customers' satisfaction, such as improving the service efficiency through training to change the probability distribution of service time; establishing online bank to lower the everyday customer flow to increase time between arrivals; we can also establish a new service window. For each solution, we discussed the minimal change to meet our goal.

At last, we gathered all our model results, and take several reasonable ones of them as the suggestion to the manager, in which way he can choose one according to the actual condition of his bank.

Introduction

The queue management system of a bank is vital to whether customers are satisfied with the bank service or not. Given the statistics of service time and time between arrivals, we used a variety of measurement techniques which predicted and measured queue lengths and waiting times and provided management information to help service levels and resource deployment. Then we determined whether the current one was satisfactory to the manager guidelines, which expected an average queue length under 2 persons and an average waiting time under 2 minutes.

To accomplish the manager's goal, we put forward several technical and non-technical solutions which required the minimal changes for servers and wrote a letter to the manager of our best solutions.

General Assumptions

- The probabilities in the two given tables are discrete ones.

  We analyzed the two tables and considered minute as our minimal time unit. In this situation, continuous probability makes no sense in our model.

- Arrival time distribution of the customers is balanced, which means there doesn't exist such peak hours.

  The common working hour of banks in China is around 8 hours a day, as is the result of our investigating the four major banks[1] in China. What's more, since in most cases, people have to ask for leave from their work to come to the bank, it's meaningless to consider such peak hours.

- The current service policy is FCFS[2].

  FCFS is the mostly used service policy which is reasonable while making our model simplified.

- There is only one service window.

  The number of windows is not mentioned in the problem. Since more windows will make the model complex, we started from only one window.

[1]The four major banks in China are the Industrial and Commercial Bank of China (ICBC), the Agricultural Bank of China (ABC), the Bank of China (BOC) and the China Construction Bank (CCB). In most cases, their working hours starts from 8 a.m. to 5p.m.

[2] First-Come, First-Served: a service policy whereby the requests of customers or clients are attended to in the order that they arrived, without other biases or preferences.

The Models

Basic Model A: A Mathematical Model

Introduction

We used five variables to describe a state, and found out the recursion formula of the states. In this way, we gave out a rigorous mathematical model to find out answers.

Assumption

- There is only one service window.

Model Description

- **The Queue**
  - **Terminology**

Terminology	Description
Queue	In computer science, a queue is a particular kind of abstract data type or collection in which the entities in the collection are kept in order and the principal (or only) operations on the collection are the addition of entities to the rear terminal position, known as enqueue, and removal of entities from the front terminal position, known as dequeue. This makes the queue a First-In-First-Out (FIFO) data structure[3].
Head	The first element in current queue, which is the Premier-came of all the elements in current queue.
Tail	The last element in current queue, which is the newest-came of all the elements in current queue.

Table 1

  - **Variables**

Variable	Description
$head$	The number of the head element in current queue
$tail$	The number of the tail element in current queue

Table 2

  - **Graphic expression of a queue**

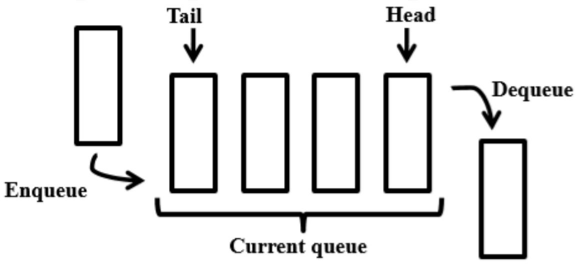

Figure 1

[3] In a FIFO data structure, the first element added to the queue will be the first one to be removed. This is equivalent to the requirement that once a new element is added, all elements that were added before have to be removed before the new element can be removed. Often a peek or front operation is also entered, returning the value of the front element without dequeuing it.

◆ Variables

Variable	Description
T	The current time, in the unit of minute
n	The number of customers in the waiting queue, including the one being served at present, which is the head of the waiting queue.
$t1$	The last arrived customer arrived $t1$ minutes ago
$t2$	The present customer being served, which is the head of the queue, needs another $t2$ minutes to be served and then leaves.
N	The number of the total served and already-came customers, which is also the number of the tail customer in the queue
$P^{T}_{n,t1,t2,N}$	The probability of having n customers in the waiting queue, the last arrived customer arrived $t1$ minutes ago, the present customer being served needs another $t2$ minutes to leave while the number of the total served and already-came customers is N at time T. ($0 \le n \le 150$ $0 \le N \le 150$, $1 \le t1 \le 5$, $1 \le t2 \le 4$)

Table 3

◆ Constants

Constant	Description
a_{t1}	The probability of some people will come when the last-came customer came $t1$ minutes ago.
CA_t	Time between arrival is t minutes.
CS_t	The service time of the head customer in the queue is t minutes.

Table 4

■ Arrival times

Time between arrival(min.)	0	1	2	3	4	5
Constant	CA_0	CA_1	CA_2	CA_3	CA_4	CA_5
Probability/Value	0.10	0.15	0.10	0.35	0.25	0.05

Table 5

■ Service Time

Service time(min.)	1	2	3	4
Constant	CS_1	CS_2	CS_3	CS_4
Probability/Value	0.25	0.20	0.40	0.15

Table 6

◆ The calculation of a_t

■ Conditional Probability

★ Terminology

Terminology	Description
P_A	The probability of event A occurs.
$P_{A \cap B}$	The probability of event A and B occur at the same time
$P_{A \mid B}$	The conditional probability of A given B, which is the probability that event A will occur when event B is known to occur or to have occurred.
+=	a+=b means a:=a+b

Table 7

★ Axiom of probability

$$P_{A \cap B} = P_{A \mid B} \cdot P_B$$

Although mathematically equivalent, this may be preferred philosophically. Since it is common knowledge in probability theory, we did not prove it in mathematics here.

★ The calculation of a_{t1}

We defined event ***A*** as: At time T some people came. And then defined event ***B*** as: ***t1*** minutes' no-one-came situation exists. In this case,

According to the axiom above, we got the following equations.

$$P_{A \cap B} = P_{A \mid B} \cdot P_B$$

That is, $P_{A \mid B} = \frac{P_{A \cap B}}{P_B}$ .... (1)

According to our definition, it is obvious that

$$P_{A \mid B} = a_{t1}$$

,which is what we wanted to solve out through the following calculations.

P_B means the probability of having waited for ***t1*** minutes for another customer to come, cases that arrival time are longer than ***t1*** are concerned, or another customer has already came some time before. In those cases, the current ***t1*** means the time between current time T and the time when the second last customer in the waiting queue came, which contradicts with the definition of ***t1.***

Therefore, we get the following equation:

$$P_B = \sum_{i=t1}^{5} CA_i \quad \text{.............} \quad (2)$$

$P_{A \cap B}$ means the probability of we have already waited for ***t1*** minutes and new customers came at current time T, that is to say, time between arrival is ***t1*** minutes, obviously, we got the following equation:

$$P_{A \cap B} = CA_{t1} \quad \text{.............} \quad (3)$$

Then put (3),(2) into (1) and finally we solved a_{t1}:

$$a_{t1} = \frac{CA_{t1}}{\sum_{i=t1}^{5} CA_i}$$

★ The value of a_{t1}

i	1	2	3	4	5
a_i	$\frac{1}{6}$	$\frac{2}{15}$	$\frac{7}{13}$	$\frac{5}{6}$	1

Table 8

◆ The recursion formula

■ The Recursion formula in most cases: $n \neq 0$

When given a state $P^{T}_{n,t1,t2,N}$,($n \neq 0$) We would like to find out all the $P^{T+1}_{n',t1',t2',N'}$ states which can be transited from $P^{T}_{n,t1,t2,N}$.

★ $t2 \neq 1$: A Simple case

Since $t2 \neq 1$ and $1 \leq t2 \leq 4$, that is to say the service has not finished yet and the customer being served the next minute will still be this customer. But we do not know how many people will come the next minute. There are two cases:

✦ No one come the next minute

Then at time ***T+1***, the number of customers in the waiting queue would still be ***n***, the number of customers already came and even left would remain to be ***N***, since time was put forward, the time that the last customer came would be ***t1+1*** and the remaining service time required for the current customer would be reduced to ***t2-1***. Since the probability of someone came is a_{t1}, the probability of no one came would be (1-a_{t1})$P^{T}_{n,t1,t2,N}$, which can be described as $P^{T+1}_{n,t1+1,t2-1,N}$

So we have:

$$P^{T+1}_{n,t1+1,t2-1,N} += (1-a_{t1})P^{T}_{n,t1,t2,N}$$

✦ Some people come the next minute

Since the probability of someone comes at time ***T*** is $a_{t1}P^{T}_{n,t1,t2,N}$, let us first look into the case that only one person comes the next minute. According to the data given, the probability of time between arrival =0 is 0.1, that is to say more than one person comes, which is not what we wanted. To have only one person come the next time, the time-between-arrival of the next customer should be more than 1min. So we got the following equations:

Probability of only one person comes at ***T+1:***

$$0.9 \cdot a_{t1} \cdot P^{T}_{n,t1,t2,N}$$

Which can be described as $P^{T+1}_{n+1,1,t2,N}$

Probability of more than one person comes at ***T+1:***

$$0.1 \cdot a_{t1} \cdot P^{T}_{n,t1,t2,N}$$

Let's take a further consideration in the situation of more than one person comes. If only two person comes, similarly like how we get the probability of only one person comes, multiply the probability of more than one person comes at ***T+1*** by a 0.1 would lead us to the result. Now let's go a step further, the probability of ***i*** people come will be

$$\mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,t2,N}}$$

Which can be described as $\boldsymbol{P^{T+1}_{n+i,1,t2-1,N+i}}$

But we should notice that ***N+i*** has a upper bound of 150. When ***N+i = 150,*** the state transition from $\boldsymbol{P^{T}_{n,t1,t2,N}}$ will be different: Since when ***N+i*** = 150, no one will come the next minute, so the situation of ***N+i+1*** will not exist. In this cases, 0.9 should not be multiplied.

So we got the following equations:

When $n \neq 0$ and $t2 \neq 1$, state $\boldsymbol{P^{T}_{n,t1,t2,N}}$ can transit to the following states:

$$\begin{cases} \boldsymbol{P^{T+1}_{n,t1+1,t2-1,N}} += (1-\boldsymbol{a_{t1}})\boldsymbol{P^{T}_{n,t1,t2,N}} \\ \boldsymbol{P^{T+1}_{n+i,1,t2-1,N+i}} += \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,t2,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{n+i,1,t2-1,N+i}} += \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,t2,N}} & \boldsymbol{N+i=150} \end{cases}$$

★ $\boldsymbol{t2 = 1}$:

In these cases, when ***T+1***, the present head of the queue has left and the service provided to a new customer starts. There are four cases: This customer needs a service of 1,2,3,4 minutes. Similarly to how we analyzed such t2≠1 and $n \neq 0$ cases, following states can be transited to:

$$\begin{cases} \boldsymbol{P^{T+1}_{n-1,t1+1,1,N}} += \mathbf{0.25} \cdot (1-\boldsymbol{a_{t1}})\boldsymbol{P^{T}_{n,t1,1,N}} \\ \boldsymbol{P^{T+1}_{n-1+i,1,1,N+i}} += \mathbf{0.25} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{n-1+i,1,1,N+i}} += \mathbf{0.25} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i=150} \\ \boldsymbol{P^{T+1}_{n-1,t1+2,1,N}} += \mathbf{0.2} \cdot (1-\boldsymbol{a_{t1}})\boldsymbol{P^{T}_{n,t1,1,N}} \\ \boldsymbol{P^{T+1}_{n-1+i,1,2,N+i}} += \mathbf{0.2} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{n-1+i,1,2,N+i}} += \mathbf{0.2} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i=150} \\ \boldsymbol{P^{T+1}_{n-1,t1+1,3,N}} += \mathbf{0.4} \cdot (1-\boldsymbol{a_{t1}})\boldsymbol{P^{T}_{n,t1,1,N}} \\ \boldsymbol{P^{T+1}_{n-1+i,1,3,N+i}} += \mathbf{0.4} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{n-1+i,1,3,N+i}} += \mathbf{0.4} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i=150} \\ \boldsymbol{P^{T+1}_{n-1,t1+1,4,N}} += \mathbf{0.15} \cdot (1-\boldsymbol{a_{t1}})\boldsymbol{P^{T}_{n,t1,1,N}} \\ \boldsymbol{P^{T+1}_{n-1+i,1,4,N+i}} += \mathbf{0.15} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{n-1+i,1,4,N+i}} += \mathbf{0.15} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{n,t1,1,N}} & \boldsymbol{N+i=150} \end{cases}$$

- **The Recursion formula in most cases: $\mathbf{n = 0}$ cases**

$$\begin{cases} \boldsymbol{P^{T+1}_{0,t1+1,1,N}} += (1-\boldsymbol{a_{t1}})\boldsymbol{P^{T}_{0,t1,1,N}} \\ \boldsymbol{P^{T+1}_{i,1,1,N+i}} += \mathbf{0.25} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{0,t1,1,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{i,1,2,N+i}} += \mathbf{0.2} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{0,t1,1,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{i,1,3,N+i}} += \mathbf{0.4} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{0,t1,1,N}} & \boldsymbol{N+i<150} \\ \boldsymbol{P^{T+1}_{i,1,4,N+i}} += \mathbf{0.15} \cdot \mathbf{0.9} \cdot \boldsymbol{a_{t1}} \cdot \mathbf{0.1}^{\boldsymbol{i-1}} \cdot \boldsymbol{P^{T}_{0,t1,1,N}} & \boldsymbol{N+i<150} \end{cases}$$

- **The Initial Value**

$$\boldsymbol{P^{0}_{0,1,1,0}} = \mathbf{1}$$

- **Data Process**

With the recursion formulas, we could figure out all the states $P^{T}_{n,t1,t2,N}$. With these data, we can figure out the following data:

$$T_{max}=754$$

The average waiting time: $\dfrac{\sum_{j=0}^{T_{max}}\sum_{i=1}^{150}[P^{j}_{i,t1,t2,N}\times(i-1)]}{150}$

The average total covered length of time: $T_{average}=\sum_{j=0}^{T_{max}}P^{j}_{0,0,1,150}\times j$

The average length of waiting queue: $\dfrac{\sum_{j=1}^{T_{max}}\sum_{i=1}^{150}(P^{j}_{i,t1,t2,N}\times i)}{T_{average}}$

Result

We used computer to calculate and got the following data:

- The average length of waiting queue: 2.2936 persons.
- The average waiting time: 4.74 minutes

  The current model doesn’t satisfy the requirement of the manager in that the average waiting time is far more than 2 minutes.

Basic Model B: A Simulation Model

Introduction

Monte Carlo Method[4] refers to those methods that repeat simulations to generate the average value, in this way to solve the problem, which is a good way to solve our problem. In model B, we built up our simulation model and got the data we need.

Algorithm Description

◆ Terminology

Terminology	Description
Queue	In computer science, a queue is a particular kind of abstract data type or collection in which the entities in the collection are kept in order and the principal (or only) operations on the collection are the addition of entities to the rear terminal position, known as enqueue, and removal of entities from the front terminal position, known as dequeue. This makes the queue a First-In-First-Out (FIFO) data structure[5].
Head	The first element in current queue, which is the Premier-came of all the elements in current queue.
Tail	The last element in current queue, which is the newest-came of all the elements in current queue.

Table 9

◆ Graphic expression of a queue

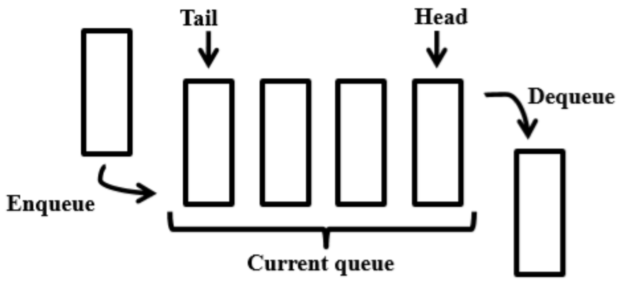

Figure 2

[4] Monte Carlo methods (or Monte Carlo experiments) are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results; i.e., by running simulations many times over in order to calculate those same probabilities heuristically just like actually playing and recording your results in a real casino situation: hence the name. They are often used in physical and mathematical problems and are most suited to be applied when it is impossible to obtain a closed-form expression or infeasible to apply a deterministic algorithm. Monte Carlo methods are mainly used in three distinct problems: optimization, numerical integration and generation of samples from a probability distribution. Monte Carlo methods are especially useful for simulating systems with many coupled degrees of freedom, such as fluids, disordered materials, strongly coupled solids, and cellular structures (see cellular Potts model). They are used to model phenomena with significant uncertainty in inputs, such as the calculation of risk in business. They are widely used in mathematics, for example to evaluate multidimensional definite integrals with complicated boundary conditions. When Monte Carlo simulations have been applied in space exploration and oil exploration, their predictions of failures, cost overruns and schedule overruns are routinely better than human intuition or alternative "soft" methods. —From Wikipedia

[5] In a FIFO data structure, the first element added to the queue will be the first one to be removed. This is equivalent to the requirement that once a new element is added, all elements that were added before have to be removed before the new element can be removed. Often a peek or front operation is also entered, returning the value of the front element without dequeuing it. From Wikipedia

◆ Variables

Variable	Description
$\boldsymbol{T}$	The current time, in the unit of minute
$\boldsymbol{i}$	We use number ***i*** to express the ***i***th customer, $\boldsymbol{1 \leq i \leq 150}$
$\boldsymbol{head}$	The number of the head element in current queue
$\boldsymbol{tail}$	The number of the tail element in current queue
$\boldsymbol{arrive_i}$	The time when customer ***i*** arrived
$\boldsymbol{leave_i}$	The time when customer ***i*** left
$\boldsymbol{serve_i}$	The left required length of time to serve customer ***i***
$\boldsymbol{l_T}$	The length of the queue at $\boldsymbol{T}$

Table 10

We put forward an algorithm, in this way we can stimulate the whole process and get the expected data.

◆ Algorithm Description

1. *Judge* ***if*** *the 150 people have all arrived* ***and*** *the queue is empty?*

    If Yes→ *Turn to* 11

    If No→ *Turn to* 2

2. $\mathrm{T} = \mathrm{T} + 1$

3. *Judge if a new customer has arrived at this moment?*

    If Yes→ *Turn to* 4

    If No→ *Turn to* 5

4. *Add the new customer* $\boldsymbol{tail + 1}$ *to the end of the queue and record the arrival time of him, which is* $\boldsymbol{arrive_i}$*. The required service time* $\boldsymbol{serve_{tail+1}}$ *is generated by the program according to the given probability*

5. *Judge if the current queue is empty?*

    If Yes→ *Turn to* 9

    If No→ *Turn to* 6

6. *Reduce the left required service time of the head element in he queue by 1, that is* $\boldsymbol{serve_{head} = serve_{head} - 1}$

7. *Judge if the service towards the head customer in the queue has ended? That is, is* $\boldsymbol{serve_{head} = 0}$?

    If Yes→ *Turn to* 8

    If No→ *Turn to* 9

8. *The head customer leaves, that is dequeue* $\boldsymbol{head}$***.*** *Record the leaving time of* $\boldsymbol{head}$***,*** *that is*

    $\boldsymbol{leave_{head} = T}$.

    $\boldsymbol{head = head + 1}$

9. *Record current length of queue* $\boldsymbol{l_T = tail - head + 1}$

10. *Turn to 1*

11. *End*

◆ Data processing

The algorithm provides the arrival time and the leaving time of every customer. With these data, we can figure out the following data:

The average length of the waiting queue = $\frac{\sum_{t=1}^{T} l_t}{T}$

The average waiting time = $\frac{\sum_{i=1}^{150}(leave_i - arrive_i)}{150}$

Program

We used *Matlab*[6] to realize our algorithm. The program is shown below(Four matlab documents are included: serve.m,cust.m, queue.m, queue_sum.m):

```
Serve.m
function x=serv();
a=randi([0,999],1,1);
if (a<250)
        x=1;
elseif (a<450)
        x=2;
elseif (a<850)
        x=3;
else
        x=4;
end;

cust.m:
function x=cust(m);
a=randi([0,999],1,1);
if (a<100-100*m)
        x=0;
elseif (a<250-150*m)
                x=1;
elseif (a<350-100*m)
                x=2;
elseif (a<700-350*m)
                x=3;
elseif (a<950-250*m)
                x=4;
elseif (a<1000-50*m)
                x=5;
else
                x=6;
end;

queue.m:
function
[len,wait,free,time]=queue(n)
;
m=0;
n=floor(n*(1-m));

head=1;
tail=0;
t1=cust(m);
maxt=5*n;
peo=zeros(1,n+2);
lenx=zeros(1,maxt);
arrivex=zeros(1,n+2);
waitx=zeros(1,n+2);
leavex=zeros(1,n+2);
freex=zeros(1,maxt);
T=0;


  while (tail<=n)
        T=T+1;
        while
(t1==0)&&(tail<=n)
                tail=tail+1;
                peo(tail)=serv();
                arrivex(tail)=T;
                t1=cust(m);
        end

        t1=t1-1;

        if (head<=tail)

peo(head)=peo(head)-1;
                if
(leavex(head)==0)

leavex(head)=T;
                end
                if (peo(head)==0)
                        head=head+1;
                end
        end

        lenx(T)=tail-head+1;
        if (lenx(T)==0)
                freex(T)=1;
        end
  end

  while (head<=tail)
          T=T+1;
peo(head)=peo(head)-1;
          if (leavex(head)==0)
                leavex(head)=T;
          end
          if (peo(head)==0)
                head=head+1;
          end
          lenx(T)=tail-head+1;
        if (lenx(T)==0)
              freex(T)=1;
        end
  end

  waitx=leavex-arrivex;


  len=mean(lenx(1:T));
  wait=mean(waitx(1:n));
  free=mean(freex(1:T));
  time=T;
  plot(1:T,lenx(1:T))

queue_sum.m
clear
clc

hold off;
max_i=10000;
len=zeros(1,max_i);
wait=zeros(1,max_i);
free=zeros(1,max_i);
for i=1:max_i

[len(i),wait(i),free(i),time(i)]
=queue(150);
       i
end

mean(len)
mean(wait)
mean(free)
mean(time)
```

Figure 3-The Program

[6] MATLAB (matrix laboratory) is a numerical computing environment and fourth-generation programming language. Developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, and Fortran.

Two examples of the statistics we simulate

◆ **One which simulates small results**

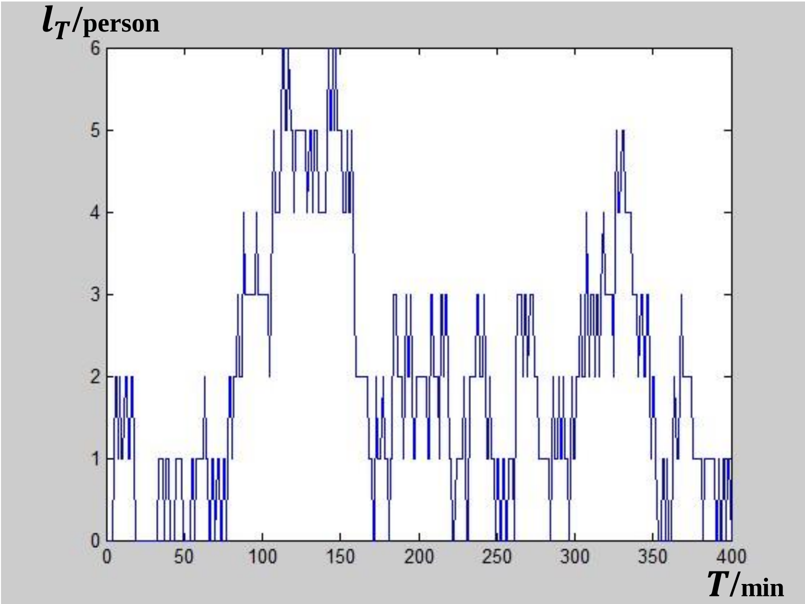

Figure 4

The average length of waiting queue: 1.99 persons
The average waiting time:3.84 minutes.

◆ **One which simulates large results**

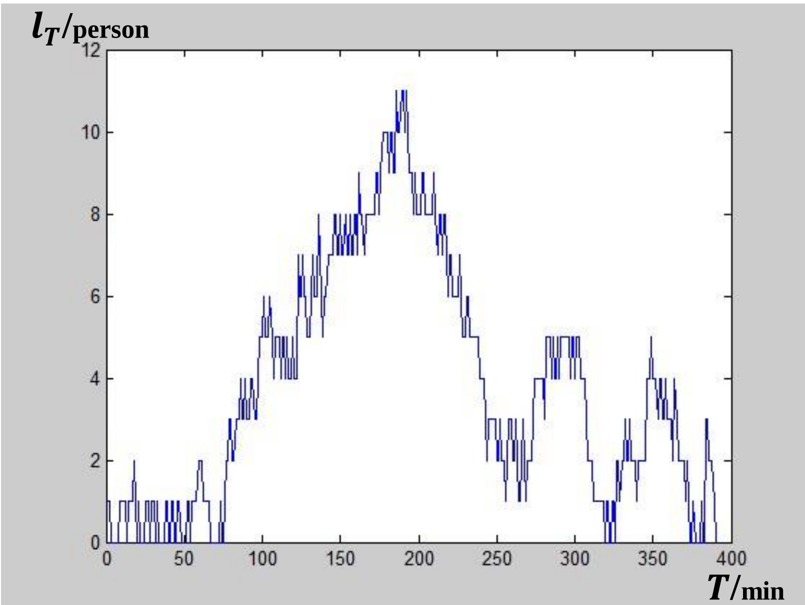

Figure 5

The average length of waiting queue: 3.84 persons
The average waiting time: 8.52 minutes.

By generating randomized data and following the algorithm we put forward, we repeat the process for 10000 times and the result we got is shown below:

Result

- The average length of waiting queue: 2.3778 persons.
- The average waiting time: 4.91 minutes

  The current model doesn't satisfy the requirement of the manager in that the average waiting time is far more than 2 minutes.

Evaluation and Comparison Between Model A and Model B

As we can see, the results we got from this two models coincided with each other.

Our Model A is one based on a recursion formula model while Model B is one based on computer simulation. The Model A is a very rigorous mathematical model which can provide accurate answers, which is its advantage. But since this model is complex and has poor portability, we almost have to change all recursions when trying to add some small changes, which are its disadvantages and adds heavy work to us. What's more, the time complexity of it is very high; therefore we only applied it in analyzing the current situation.

Meanwhile, Model B is one based on computer simulation. It's an easy-understanding one of high portability, showing what it wants to do and how it does clearly, which is its advantages and provides great convenience to us. However, it's not a rigorous mathematical model and needs huge amounts of simulation to improve its accuracy, which is its disadvantage.

Since Model B is much more portable, we decided to use Model B in the following part of this paper.

Minimal Changes

Introduction

We put forward 4 solutions to enhance the service efficiency.

Solutions

- **A suggestion: A non-technical solution but must be useful**

  Offering services like drinks, newspapers can be a very useful and satisfactory according to several psychological researches on queuing psychology. In this way, the current situation of an average waiting time for 4.9 minutes can be more satisfactory according to the customers.

- **Solution A: More service windows**

  We used computer to simulate the situation of 2 service windows. The algorithm of it is based on the one in Model B, the difference between them is that the condition of dequeuing is changed.

  - **Program**

serve.m

```
function x=serv();
a=randi([0,999],1,1);
if (a<250)
    x=1;
elseif (a<450)
    x=2;
elseif (a<850)
    x=3;
else
    x=4;
end;
```

cust.m:

```
function x=cust(m);
a=randi([0,999],1,1);
if (a<100-100*m)
    x=0;
elseif (a<250-150*m)
        x=1;
elseif (a<350-100*m)
        x=2;
elseif (a<700-350*m)
        x=3;
elseif (a<950-250*m)
        x=4;
elseif (a<1000-50*m)
        x=5;
else
        x=6;
end;
```

double_queue.m

```
%function [len,wait,free,time]=double_queue(n);
n=150;
m=0;
n=floor(n*(1-m));

head=1;
tail=0;
t1=cust(m);
maxt=5*n;
peo=zeros(1,n+2);
lenx=zeros(1,maxt);
arrivex=zeros(1,n+2);
waitx=zeros(1,n+2);
leavex=zeros(1,n+2);
freex=zeros(1,maxt);
T=0;


  while (tail<=n)
    T=T+1;
    while (t1==0)&&(tail<=n)
        tail=tail+1;
        peo(tail)=serv();
        arrivex(tail)=T;
        t1=cust(m);
    end

    t1=t1-1;

    if (head<=tail-1)
        peo(head)=peo(head)-1;
        peo(head+1)=peo(head+1)-1;
            if (leavex(head)==0)
                leavex(head)=T;
            end
            if (leavex(head+1)==0)
                leavex(head+1)=T;
            end
            if (peo(head)==0)&&(peo(head+1)==0)
                head=head+2;
            elseif (peo(head)==0)&&(peo(head+1)~=0)
                head=head+1;
            elseif (peo(head)~=0)&&(peo(head+1)==0)
                    %swap
                temp=peo(head);peo(head)=peo(head+1);peo(head+1)=temp;
                temp=leavex(head);leavex(head)=leavex(head+1);leavex(head+1)=temp;
                temp=arrivex(head);arrivex(head)=arrivex(head+1);arrivex(head+1)=temp;
                head=head+1;
            end
    elseif (head<=tail)
        peo(head)=peo(head)-1;
            if (leavex(head)==0)
                leavex(head)=T;
            end
            if (peo(head)==0)
                head=head+1;
            end
    end

    if (tail-head+1>=2)
        lenx(T)=tail-head+1-1;
    elseif (tail-head+1==1)
        lenx(T)=1;
    else
        lenx(T)=0;
    end
    if (lenx(T)==0)
        freex(T)=1;
    end
  end

  while (head<=tail)
     T=T+1;
     if (head<=tail-1)
        peo(head)=peo(head)-1;
        peo(head+1)=peo(head+1)-1;
            if (leavex(head)==0)
                leavex(head)=T;
            end
            if (leavex(head+1)==0)
                leavex(head+1)=T;
            end
            if (peo(head)==0)&&(peo(head+1)==0)
                head=head+2;
            elseif (peo(head)==0)&&(peo(head+1)~=0)
                head=head+1;
            elseif (peo(head)~=0)&&(peo(head+1)==0)
                    %swap
                temp=peo(head);peo(head)=peo(head+1);peo(head+1)=temp;
                temp=leavex(head);leavex(head)=leavex(head+1);leavex(head+1)=temp;
                temp=arrivex(head);arrivex(head)=arrivex(head+1);arrivex(head+1)=temp;
                head=head+1;
            end
     elseif (head<=tail)
        peo(head)=peo(head)-1;
            if (leavex(head)==0)
                leavex(head)=T;
            end
            if (peo(head)==0)
                head=head+1;
            end
     end

     if (tail-head+1>=2)
        lenx(T)=tail-head+1-1;
     elseif (tail-head+1==1)
            lenx(T)=1;
     else
            lenx(T)=0;
     end
     if (lenx(T)==0)
            freex(T)=1;
     end
  end

  waitx=leavex-arrivex;


  len=mean(lenx(1:T));
  wait=mean(waitx(1:n));
  free=mean(freex(1:T));
  time=T;
  plot(1:T,lenx(1:T),'-');
```

double_queue_sum.m

```
clear
clc

hold off;
max_i=10000;
len=zeros(1,max_i);
wait=zeros(1,max_i);
free=zeros(1,max_i);
for i=1:max_i
[len(i),wait(i),free(i),time(i)]=double_queue(150);
    i
end

mean(len)
mean(wait)
mean(free)
mean(time)
```

- **Result**

With two service windows working at the same time, we got the following data:

Figure 8

The average length of the waiting queue: 0.5083 people.
The average waiting time: 0.1136 minutes.
The average covered length of time: 402.618 minutes.

As is shown above, one more service window can perfectly meet the manger's requirement. Meanwhile, we should consider the cost of adding a service window, including the salary of the extra clerk, device expenses, etc. If the manager can afford these changes, it would be an excellent solution.

- **Several Simulated Examples**

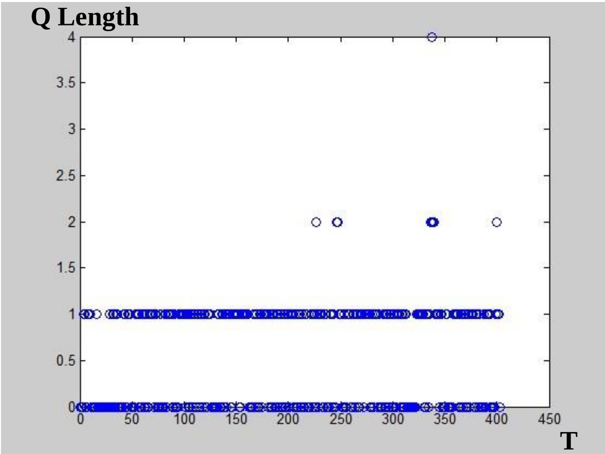

Figure 9

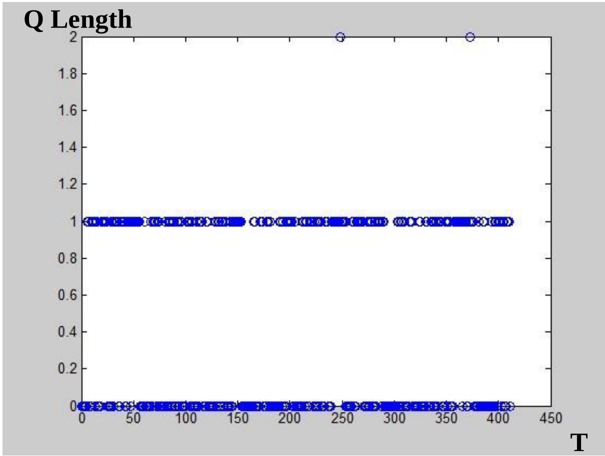

Figure 10

- **About the Result**

The expected arrival time is 2.65 minutes while the expected service time is 2.45 minutes according to the original given data. Since they were very close to each other, it's very easy for the bank to fall into the situation of "Accumulating customers". In this way, the waiting time increased greatly.

However, when two service windows are available, such "Accumulating customers" situations reduces sharply, that is why the result of two-service-window differs from only-one-service-window greatly.

◆ Solution B: Offering online service

Offering online services can finish some work beforehand such as filling up forms, in this way the amount of everyday customers is reduced so the time between arrivals is increased overall.

● The Original Data

Time between arrival(min.)	0	1	2	3	4	5
Probability/Value	0.10	0.15	0.10	0.35	0.25	0.05

Table 13

With the establishment of online bank, we assume that the customer flow is reduced to (1-m) of the original one, since the customer flow should be distributed to those arrivals between 0,1,2,3,4,5 minutes according to their original proportion, we assume that this reduced probability would be add to next minute's probability, that is:

● The New Data

Time between arrival(min)	Probability
0	$\mathbf{0.1(1-\boldsymbol{m})}$
1	$\mathbf{0.15(1-\boldsymbol{m})+0.1\boldsymbol{m}}$
2	$\mathbf{0.1(1-\boldsymbol{m})+0.15m}$
3	$\mathbf{0.35(1-\boldsymbol{m})+0.1m}$
4	$\mathbf{0.25(1-\boldsymbol{m})+0.35m}$
5	$\mathbf{0.05(1-\boldsymbol{m})+0.25\boldsymbol{m}}$
6	$\mathbf{0.05\boldsymbol{m}}$

Table 14

● The Result

We enumerated the smallest $\boldsymbol{m}$ which meets the manager's requirement of an average queue length under 2 persons and an average waiting time under 2 minutes. The result is shown below:

m = *0.34*

The average length of waiting queue: 1.1167 people.
The average waiting time: 1.9015 minutes
The average covered total length of time: 300.8 minutes.

Solution C: Improving the Service Efficiency by investing money

In this solution, we intend to change the service efficiency by investing money, that is to say, change the service time probability when ***M*** thousand dollars are invested.

The Original Data

Service Time

Service time(min.)	1	2	3	4
Probability/Value	0.25	0.20	0.40	0.15

Table 15

Qualitative Analysis

When M is small

When ***M*** is small, with the growth of invested money, the probability of 4-minute-service reduces while those 1(2,3)-minute-service increases. What's more, the 3-minute enjoys a faster growth than that of 1 or 2.

When M is large

When ***M*** is large, with the growth of invested money, even the probability of 3-minute-service also reduces while those 1(2)-minute-service increases. What's more, the 3-minute-service enjoys a faster reduction than that of 4-minute-service; The 1-minute-service enjoys a faster growth than that of 2-minute-service.

Quantitative Analysis

Now we'd like to express this process quantitatively. We assume that ***M*** ranges from 0 to 10. We put forward the following equations:

$$P_1 = \begin{cases} 0.25 + 0.002\,M, & 0 \le M \le 1 \\ 0.231 + 0.021M, & 1 \le M \le 10 \end{cases}$$
$$P_2 = \begin{cases} 0.2 + 0.003\,M, & 0 \le M \le 1 \\ 0.189 + 0.014M, & 1 \le M \le 10 \end{cases}$$
$$P_3 = \begin{cases} 0.4 + 0.01\,M, & 0 \le M \le 1 \\ 0.43 - 0.02M, & 1 \le M \le 10 \end{cases}$$
$$P_4 = 0.15 - 0.015M \qquad 0 \le M \le 10$$

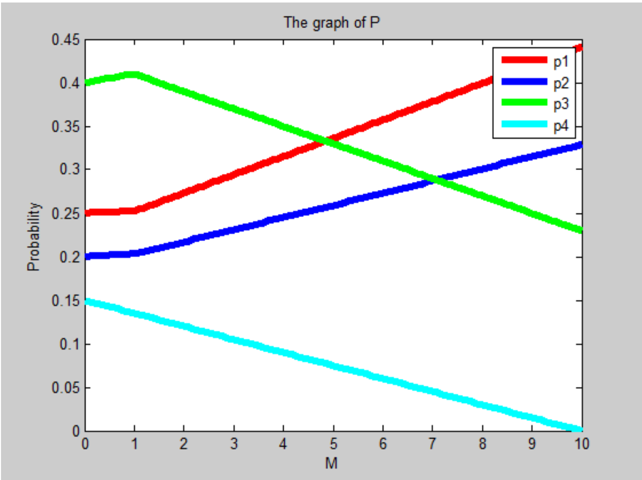

Figure 11

- **Result**

We use computer to simulate and found that when ***M*=5，** it meets the manager's requirement of an average queue length under 2 persons and an average waiting time under 2 minutes. The result is shown below:

- **Service Time**

Service time(min.)	1	2	3	4
Probability/Value	0.336	0.259	0.33	0.075

Table 16

The average length of waiting queue: 1.1552 people.
The average waiting time: 1.9174 minutes
The average covered total length of time: 404.1235 minutes.

- **Tips**

This result is not very accurate. When the probability of service is around what is given above, it would be enough to meet the manager's requirement.

Sensitivity Analysis

——Sensitivity Analysis of the number of customers

- **About the length of queue**

Amount of customers(person)	50	100	150	200	250	300
Average length of queue(person)	1.78	2.16	2.37	2.51	2.59	2.65

Table 11

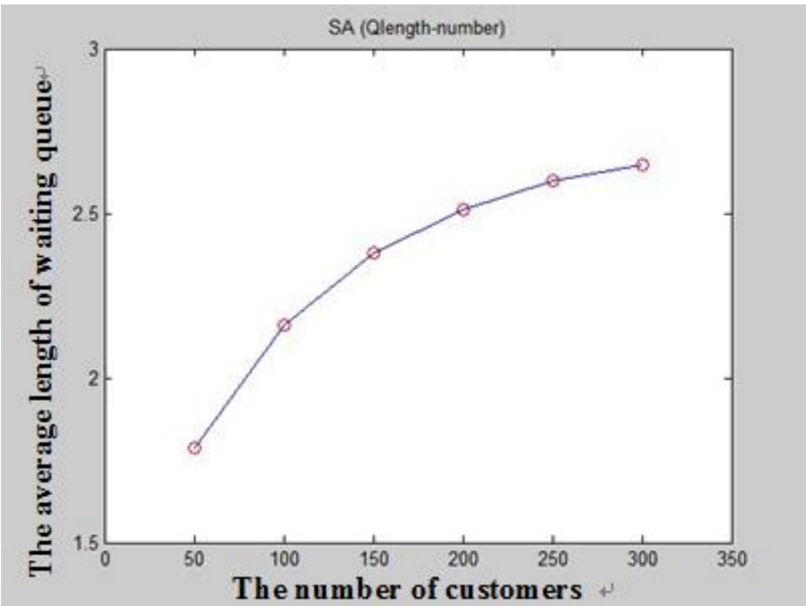

Figure 6

- **About the waiting time**

Amount of customers(person)	50	100	150	200	250	300
Average waiting time(min)	3.45	4.38	4.91	5.25	5.46	5.59

Table 12

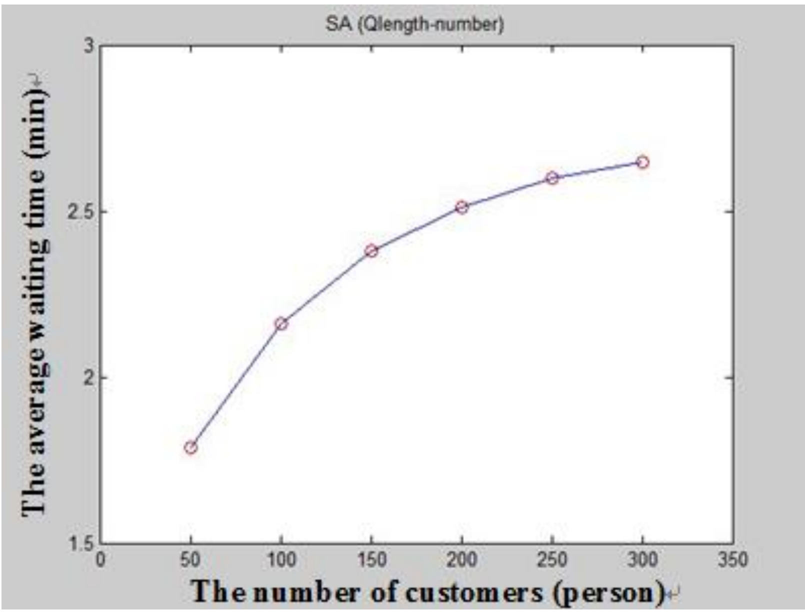

Figure 7

Conclusion

We can infer from the two graphs that when ***n*** is small, the impact on the average length of queue and the average waiting time is obvious. When ***n*** approaches infinite, the impact gradually decreases. Since the number we considered 150 is a small number, it do has some impact on the result.

Strengths and Weakness

◆ Strengths

- Model A: High Mathematical Rigor

In *Model A*, we gave the complete process of how we infer the recursion formulas and provided a very detailed classification of all the states. *Model A* enjoys a high mathematical rigor, which is important in mathematical modeling.

- Model B: Strong Portability and Clearness

In *Model B*, we gave a detailed description of our algorithm used to stimulate the process. Since *Model B* is a Monte Carlo Method, generating randomized data and figuring out our objective data for huge amounts of time can improve the accuracy of it greatly. *Model B* is a highly portable one which played an extremely important role in part *Minimal Changes* in our paper. Its easy- understanding and easy-maintaining brought great convenience to us.

- Model A and B:Consistence

The result we got from the mathematical model and the simulation model coincided with each other, this provided solid evidence that our models are correct.

- Model A and B: Universality

Since our models are universal ones based on general data, any given probability distribution of arrival time and service time can figure out a reasonable answer by using our models.

- Solutions: Diversity

In part *Minimal Changes* in our paper, we put forward several technical and non-technical solutions which look into the problem from different angles. All of them improved the current situation to some extent. We gave detailed analyses on all of them, in this way, our solutions are clear and rigorous.

◆ Weakness

- Model A: Low Portability and High Time Complexity

Since *Model A* exhausted all the possible states, the high time complexity of it and the complicated situations kept us from applying it to variable situations. Thus, such a rigorous model is wasted, which is a pity.

- Number of Service Windows

In fact, at the very start, we considered the amounts of window as a very important factor. However, since making number of service windows variable would make it almost impossible for us to build the model, we only considered the situation of one window in building the calculating model, which is our weakness.

- Discrete Probability

Since the probability given in the problem was discrete, our model is based on discrete probability, which is not very suitable for those continuous probability queueing problems.

Reference

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms. *MIT press.* 1990 (first edition).
2. Michael Mitzenmacher and Eli Upfal. Probability and Computing. *Cambridge University Press.* 2005-01-31.
3. Giuliano Casale, Ningfang Mi, and Evgenia Smirni. Bound Analysis of Closed Queueing Networks with Nonrenewal Service. *SIGMETRICS Perform. Eval. Rev.* June 2008.
4. YIN Chao, PANG Ji-ming, QI Jun-hong. The analysis and design of bank queuing business simulation system based on queue structure. *Journal of Shandong University of Technology (Sci & Tech).* Nov.2006.
5. Qiang Guo, Jianguo Liu, and Xingwen Chen. Optimization model And Simulation of The Queuing System With Quickpass. *The Sixth World Congress on Intelligent Control and Automation.* October 2006.
6. PENG Ping, SUN Shui-ling. Compute simulation of bank client service based on priority queue. *Computer Engineering and Design.* Nov. 2007.

The Letter

Dear bank management

Hello. We're a group of students who loves mathematical models. We knew about the arrival and service times of your bank as well as your requirement that the average waiting time should be less than 2 minutes and the average length of queue should be less than 2 persons. As a result, we build a model about your bank. Unfortunately, we find that the average waiting time is about 4.9 minutes and the average length of queue if about 2.4 people.

In order to meet your requirement, we provide four solutions.

1. Since your ultimate model is to improve customer satisfaction, you may provide the customers with different ways to entertain so that they may feel not so depressed to wait for 4.9 minutes.
2. If your bank is big enough, you may establish a new service window. One extra window is enough, thus reducing the average waiting time and the average length of queue dramatically, to about 0.11 minutes and 0.51 people.
3. You may also establish online bank service if possible so that some customers will choose the online service, thus lowering the everyday customer flow and also increase time between arrivals. Judging from our models, you have to make about 35% of your customers choose online service. That is about 51 customers.
4. The bank clerk may be trained to improve the service efficiency. It can meet with your requirement if you're able to finish 50% of the service which used to take up 4 minutes within 3 minutes, 36% of the service which used to take up 3 minutes within 2 minutes and 15% of the 2-minute service within 1 minute. Of course, other similar changes will also make sense.

Of course, not all solutions are fit for your bank. You can choose one solution, or you may use the solutions together according to your bank's situation. It's best only when it fits you.

Yours sincerely,
A group of HiMCM lovers.