餐饮店排队最佳预留时间的蒙特卡洛模拟

倪琳郁 10113102

2016 年 1 月 9 日

摘要 排队模型是离散事件系统最典型的的问题之一，在排队理论的基础上，结合蒙特卡洛模型能很好的模拟出某一事件在复杂情况下的平均概率。所以，蒙特卡洛为离散时间提供了很好的随机数列的产生方法。在目前的市面上，用餐排队问题越来越人性化，很多餐饮店提出某一时间段内若未能提供座位则餐费全免的政策以留住更多顾客。本文就借用门特卡洛模型，通过对相关参数的控制，解决一定情况下餐饮店的开设桌数与顾客等待时间的关系，并给出最佳的预留时间，是的餐饮店的政策在服务客人的同时能够利益最大化。

关键词 排队理论，蒙特卡洛模拟，最佳预留时间

一、引言

目前，在小资情调的餐营业中，越来越多的群众愿意排队等候就餐，这很大程度上得益于越发周到的服务政策，甚至有餐饮店提出“若 15 分钟内未能提供座位，餐费全免”的策略。那么，类似于这样的 15 分钟到底是不是符合实际情况，15 分钟外餐费全免会不会使得该店老板赔本的疑问可以很好地排队模型给出简单的证明。该文的主要写出了一般情况下的蒙特卡洛模拟，然后就简单的问题进行了讨论。

二、模型的建立

在建立模型之前，先提出两个假设：

1、假设餐饮店每天只营业 4 个小时（早上 10 点~下午 2 点），因为一般餐饮点在清早和下午是不营业的，符合实际要求。其次，晚上营业的情况与中午营业的问题分析类似，不需要重复考虑。

2、假设每桌顾客的用餐时间在 30min~1hr30min 之内，每一桌顾客看做一个整体，作为一个变量。但是这样就存在问题，所以考虑到每一家餐饮店内会有不同的桌型，每一个前来用餐的顾客单元人数是不确定的，需要建立桌型与就餐顾客单元的人数的联系并分类讨论。但是，为了简化模型，假设每一种桌型只能坐固定的人数，这样也可以达到利益的最大化，并且模拟的时候也可以把三种情况完全分开考虑。在这篇文章中，我么假设有两种桌型：4 人桌以及 8 人桌，各桌数为 a,b。

用蒙特卡洛模型模拟需要分三步走，首先要创造出理想的环境，设置参数变量并赋值，我们的参量有顾客到达时间 T(i)，顾客的用餐时间 D(i)以及就餐顾客单元与桌型的对应关系；然后计算出所需要的统计参量，在此模拟中，我们需要计算的参量有顾客的等待时间 W(i), 等待队长 WQ(i)，顺便我们也会给出队长 Q(i)和逗留时间 ST(i);最后，我们就所得到的相关量重复多次求平均值，得到 n 天内的平均等待时间，平均队长,在此文中 n 取 100。

（一）赋值程序

（1）到达时间 Arrivetim

由假设可知，到达时间是一组由范围在（0,4）之间的时间值。根据实际情况，在中午 12 左右就餐人数会达到峰值，所以我们假设整 12 点是峰值处，即 2 处是峰值，分布类似于均值为 2，方差为 1 的正态分布。但是要注意产生的随机数必须在（0.4），并且为了之后的计算方便，将数组按从小到大排序。

```
function [T] = Arrivetime( totalnum)
%totalnum是总顾客人数
%mean是产生随机数的种子
d=normrnd(2,1)
while(d<0|d>4)
    d=normrnd(2,1);
end
t(1)=d;
for(i=2:1:totalnum)
    d=normrnd(2,1);
    while(d<0|d>4)
        d=normrnd(2,1)
    end
        t(i)=d;
end
T=sort(t);
end
```

（2）就餐时间 Diningtime

根据假设 2，用餐时间设在（0.5,1.5）的时间范围内，过程类似于到达时间 Arrivetime,就不再重复。

```
function [D] = Diningtime(T)
%UNTITLED2 Summary of this function goes here
% T服务的总人数
w=rand()*2+0.5
for i=1:1:length(T)
    D(i)=w;
    w=rand()*0.5+0.5;
end
end
```

（3）顾客人数与对应桌型

就餐总顾客单元数是总量 totalnum,判断该顾客单元应该选择哪种桌型。

```
function [S] = Sort(totalnum,totaldesk)
%   totalnum是吃饭的总顾客单元数
%   totaldesk是某种桌型总数
for j=1:1:totaldesk
    S(j)=0;
end
for i=1:1:totalnum
    d=randi([1,totaldesk])
    for k=1:1:totaldesk
        if(d==k) S(k)=S(k)+1;
        end
    end
end
end
```

（二）求解程序

（1）等待时间 Waittime

只看一种情况，二人桌有 a 张，刚开业，前来就餐的顾客单元为两人的数目等于桌数时 a 时，这 a 个顾客不用等待，可直接就餐，即 W(1~a)=0;但是，当 a+1 个顾客到达时，此顾客单元为四人人，那么需要检查 a 个桌子是否有人离开，若有人离开，则 W(a+1)=0,若没有人离开，则 W(a+1)=（a 桌中最早离开时间-第 a+1 个顾客到达时间）。

```
function [W] = Waittime( S,I,c )
% I是到达时间矢量
% S是上桌时间矢量
% c是某桌型的桌数
for i=1:1:c
W(i)=0;
temW(i)=0;
end
for i=c+1:1:length(S)
    for j=1:1:c
        temp(j)=I(i-j)+S(i-j)+temW(i-j);
    end
    tem=min(temp);
    if(I(i)<tem)
        temW(i)=tem-I(i);
        W(i)=tem-I(i);
    else
        temW(i)=0;
        W(i)=0;
    end
    for j=1:1:c
        if(tem==temp(j))
            S(i-j)=S(i-c);
            I(i-j)=I(i-c);
            temW(i-j)=temW(i-c)
        end
    end
end
end
```

（2）等待队长 Waitqueuelength

为了简化此过程，可以设两个参考量，正在就餐的人数 In 和已经用完餐离开的人数 Out。已知前 a 个人的等待队长 WQLen=0；若等待时间 W（a+1)=0,显然该顾客的等待队长为零，否则用第 a+1 个人的到达时间与前 a 个人的 In 和 Out 量比较，从而得到当 a+1 位顾客到达时有多少人离开，有多少人还在就餐，WQLen(a+1) =（a+1)-（此时还在就餐人数-此时已经离开人数）-1。

```
function [ WQLen ] = Waitqueuelength( D,I,W,c )
% I是到达时间矢量
% D是就餐时间矢量
% W是等待时间矢量
% c是某桌的桌数
for i=1:1:length(D)
    In(i)=I(i)+W(i);
    Out(i)=In(i)+D(i);
end
for i=c+1:1:length(D)
    CountOut=0;
    CountIn=0;
    if(W(i)==0)
        WQLen(i)=0;
    else if(0==0)
        for j=1:1:i-1
            if(I(i)>Out(j))
                CountOut=CountOut+1;
            else if(I(i)>In(j))
                    CountIn=CountIn+1;
                end
            end
        end
        WQLen(i)=i-1-CountIn-CountOut;
        end
    end
end
end
```

(3)停留时间 Staytime

停留时间=等待时间+就餐时间

```
function [SI] = Staytime( D,I,W )
% I是到达时间矢量
% S是服务时间矢量
% W是等待时间矢量
for i=1:length(D)
    if(W(i)==0)
        SI(i)=D(i);
    else
        SI(i)=W(i)+D(i);
    end
end
end
```

(4)队长 Queuelength

Q=等待队长+相应桌数

```
function [ Q ] = Queuelength( S,I,W,WQLen,c)
% I是到达时间矢量
% S是服务时间矢量
% W是等待时间矢量
% WQLen是等待队长
% c是相应桌数
for i=1:length(W)
    In(i)=I(i)+W(i);
    Out(i)=In(i)+S(i);
end
for i=1:length(S)
    if(W(i)==0)
        Q(i)=0;
        for j=1:i-1
            if(I(i)<Out(j))
                Q(i)=Q(i)+1;
            end
        end
    else
        Q(i)=WQLen(i)+c;
    end
end
end
```

(5)数据保存 Save

```
function [ Save ] = Save( SS,totlenum,jj)
tmp=strcat('SS(I,S,W,SI,WQ,Q)',num2str(jj),'.txt')
fid=fopen(tmp,'w')
for i=1:1:6
    for j=1:1:totlenum
        if j==totlenum
            fprintf(fid,'%g\n', SS(i,j));
        else if j~=totlenum
            fprintf(fid,'%g\t',SS(i,j));
            end
        end
    end
end
fclose(fid);
end
```

三、数据处理和分析

桌数	等待时间	等待队长
5	5.55287	36.6122
6	4.35176	34.233
7	3.50303	32.0614
8	2.85918	30.0126
9	2.29743	27.1266
10	1.92032	25.0576
11	1.59924	22.8646
12	1.39165	21.8134
13	1.19543	20.0922
14	0.969448	17.6468
15	0.845096	16.3294

Fig.8 一种桌数情况

第二种桌数 \ 第一种桌数	5	6	7	8	9	10
5	0.708708	0.687717	0.81503	1.10159	1.1497	1.1288
6	0.97528	0.750384	0.706268	0.592248	1.22068	0.711528
7	0.59932	0.445893	0.480834	0.53985	0.535185	0.715984
8	0.162582	0.473409	0.339472	0.475228	0.473076	0.267358
9	0.364528	0.244574	0.221934	0.169695	0.298224	0.343075
10	0.246144	0.161853	0.433339	0.153582	0.275528	0.155456
11	0.192286	0.0831108	0.113182	0.112203	0.126559	0.0583332
12	0.120506	0.124914	0.0477889	0.0892112	0.0974369	0.0554656
13	0.066354	0.0519498	0.0887798	0.06681	0.0381643	0.138214
14	0.0621593	0.0380517	0.015569	0.0366929	0.0191767	0.0147366
15	0.061144	0.0203884	0.02566	0.0135198	0.0544662	0.0354487

Fig.9 两种桌型的情况

假设顾客与桌子的种类一一对应，也就是说即使 8 人桌有空位，人数为 4 的顾客单元也不能先坐到 8 人桌，这个假设的前提是在用餐高峰期保证效益最大化。

当只有一种桌型时，桌数为 8，以 100 次为一个循环周期求平均值。fig.1 显示的是等待时间随顾客人数的变化，可以看出，曲线可以较好地拟合为一次曲线，随着顾客人数的增多，等待时间也在不断变大。Fig.2 是等待人数和等待队长的曲线，同样可以很好的拟合成一条直线， 这与现实情况相符合。

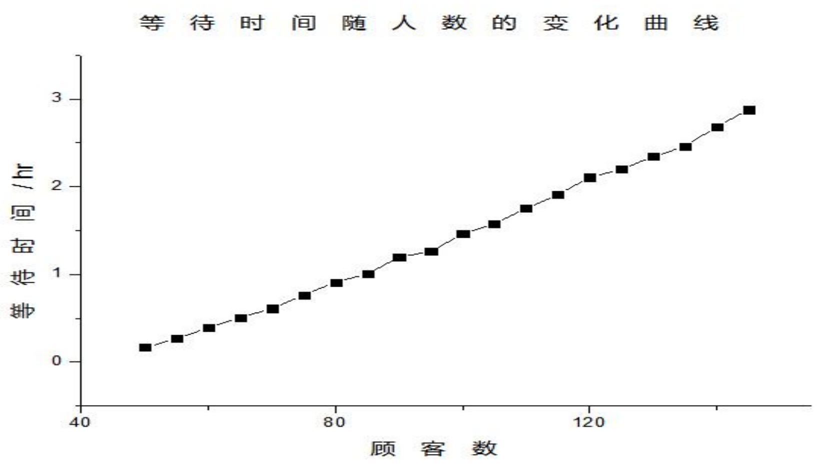

Fig.一种桌型时等待时间随顾客数的变化

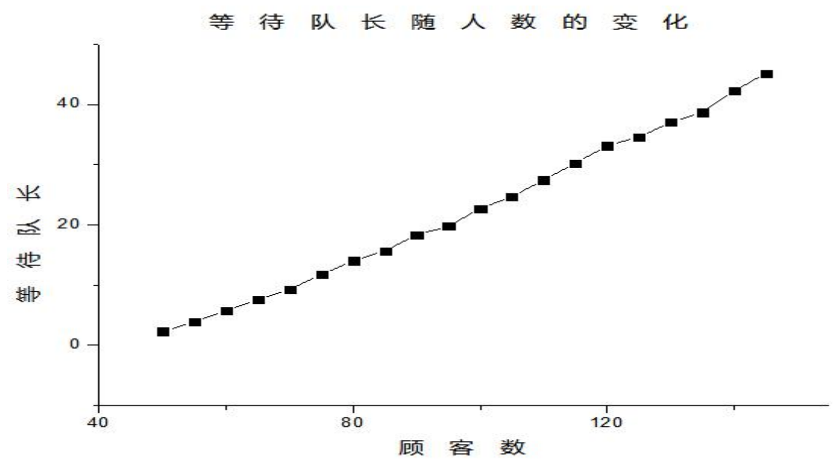

Fig.2 一种桌型等待队长随顾客数的变化

但是，如果餐饮店里的桌子种类大于一种，其曲线会很不一样，简单分析一下两种桌子的情况。

假设该餐饮店有两种桌子，第一种桌子的桌数在 5~10 之间，第二种桌子的桌数在 5~15 之间，fig.3 是第二种桌子不变第一张桌子从 5 增长到 10 的变化曲线,fig.4 是第一张桌子不变第二张桌子从 5 增长到 15，比较这两幅图，fig.4 的 6 条曲线区分不明显，fig.3 的 11 曲线区分度相比而言很明显。

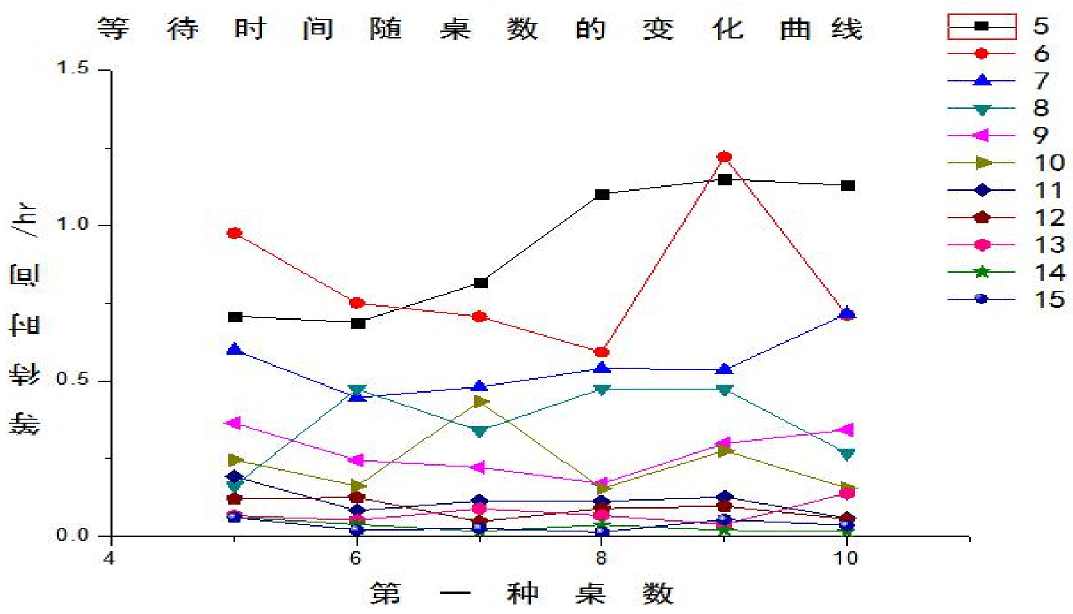

Fig.3 两种桌型其中第二种桌型数不变的情况

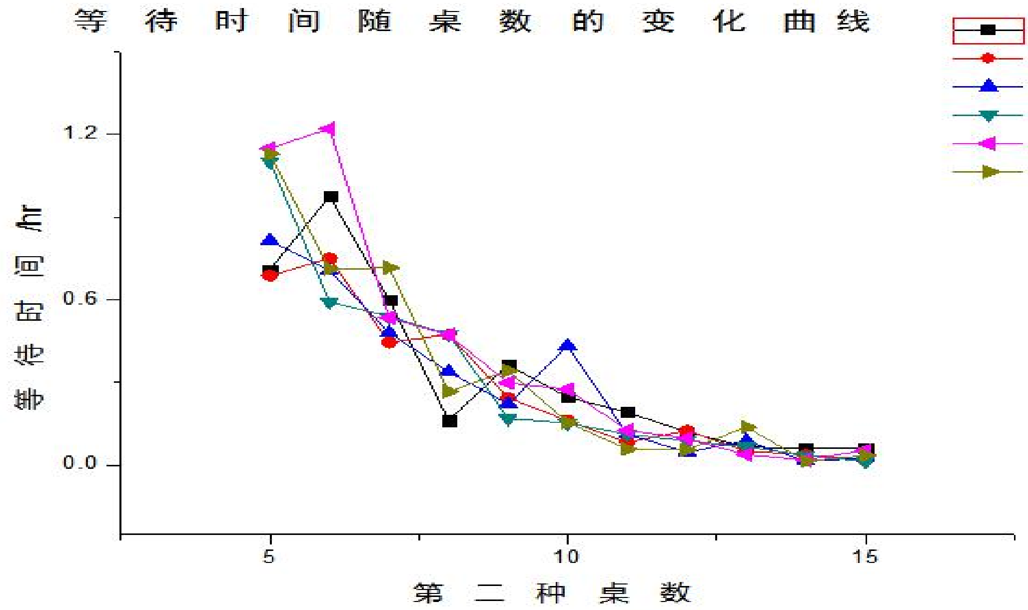

Fig.4 两种桌型其中第一种桌型数不变的情况

但是，从 fig.4 中可以看出在第二种桌数为 9 之后，曲线变化趋于平缓，但总体趋势依旧是逐渐靠近 x 轴，通过对照实验数据，可知此时的平均等待时间接近于 15min,等待队长接近于 3 个单位顾客，这与一般餐饮店中的预留等待时间一致。分析 fig.3,在第二种桌子的桌数达到 9 以后的曲线趋于平缓，这种情况与从 fig.4 里得到的结果是一致的。现在比较 fig.5 和 fig.6。fig.5 是只有一种桌子时等待时间随桌数的变化曲线。fig.6 是两种桌子情况时，第二种桌子分别为 5,6,7,8 的曲线。Fig.5 的曲线可以近似拟合为双曲线，在 fig.7（一种桌子是等待队长随桌数的变化曲线）可以拟合为直线。在 Fig.6 中，当第二种桌数为 5 时，曲线不是

单纯的下降，而有明显上升的趋势。所以，当从一种桌子变为两张桌子是，等待时间随桌数的变化不再是单调递减的，情况比较复杂，蒙特卡洛模拟为这种情况提供了很好的模拟方案，甚至可以用来解释三种或者更多种桌型的情况。

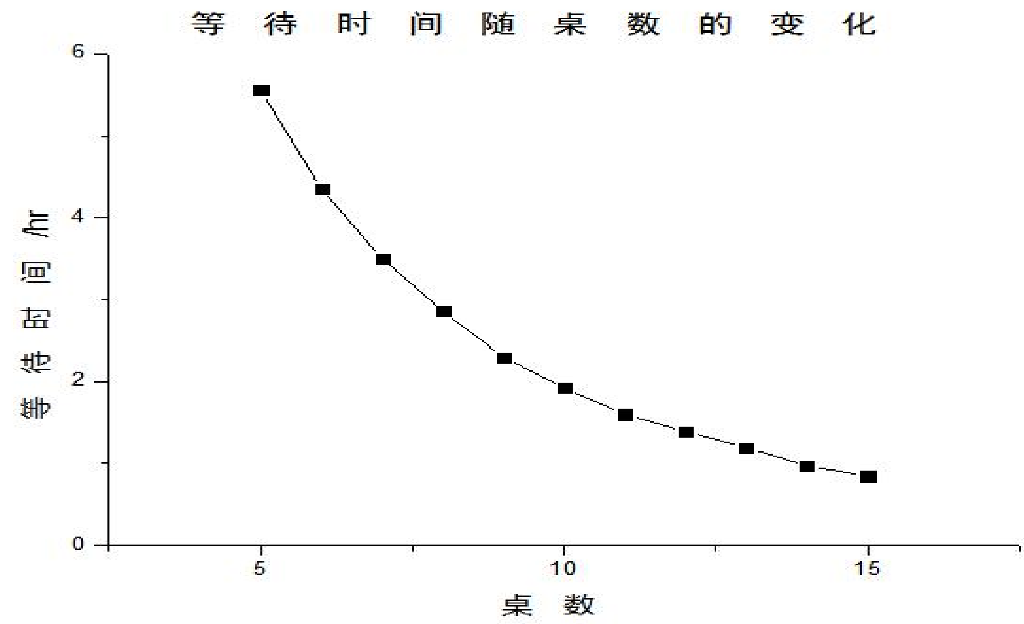

Fig.5 一种桌型等待时间随桌数的变化曲线

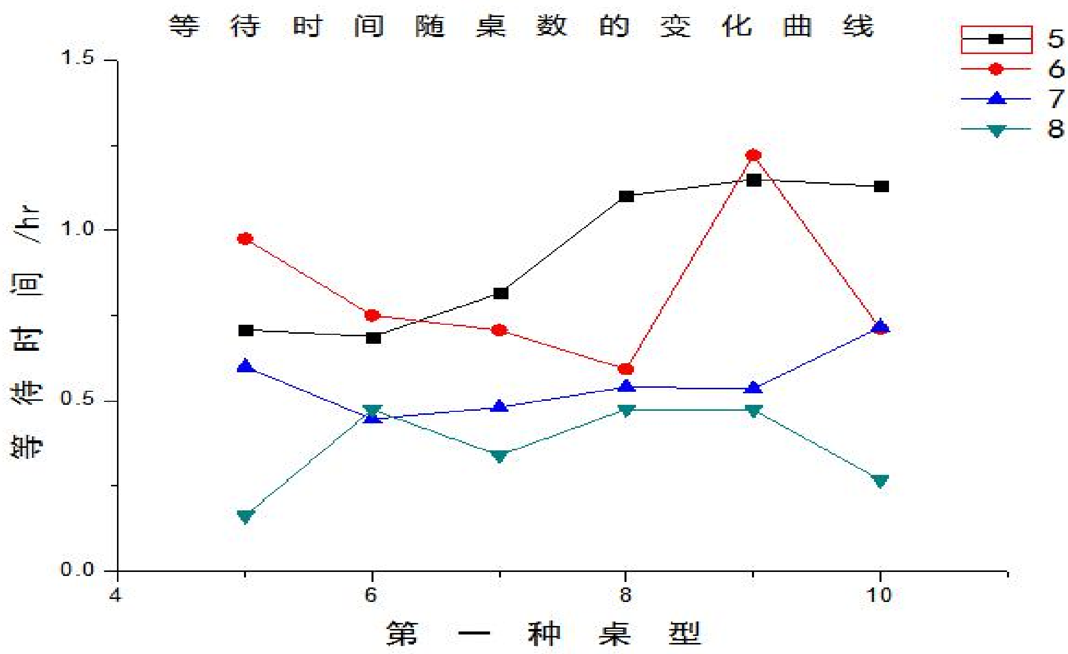

Fig.6 两种桌型第二种桌数不变的曲线

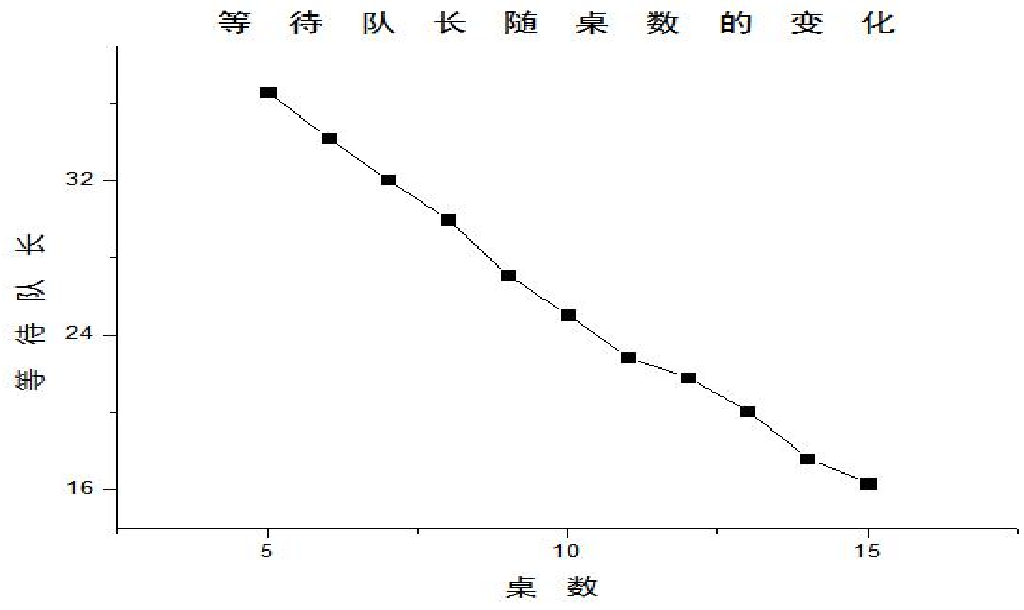

Fig.7 一种桌型等待队长随桌数的变化曲线

四、总结

综合以上的分析，一般餐饮店里规定的等待时间——15min 是合理的。以两种桌型 A、B 为例，通过控制变量法分析，可以看出在一定桌数之后，时间曲线随桌数的变化趋势趋向于平缓，而这个点之后的等待时间小于 15min。但是，等待时间的确定与桌数和桌型的种类数有关，还是应该具体问题具体分析，而蒙特卡洛模拟具有普适性，可以应用于多种情况的讨论。

五、参考文献

1、吴庆标，一类排队系统模型的计算机模拟[I],计算机应用于软件，1993
2、F.hayes—Roth etal, Building Expert System Addition—Wesley—publishing,1983
3、韩瑞峰等，排队模型的计算机模拟，2006
4、吴庆标，多顾客流排队系统的计算机模拟，1994