第八届“认证杯”数学中国

数学建模国际赛

承　诺　书

我们仔细阅读了第八届“认证杯”数学中国数学建模国际赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。我们知道，抄袭别人的成果是违反竞赛规则的,

如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们允许数学中国网站(www.madio.net)公布论文，以供网友之间学习交流，数学中国网站以非商业目的的论文交流不需要提前取得我们的同意。

我们的参赛队号为：　4321

我们选择的题目是：　A

参赛队员 (签名) ：

队员1:　P. Smith 

队员2:　小李

队员3:　小王

参赛队教练员 (签名)：　老师

第八届“认证杯”数学中国

数学建模国际赛

编 号 专 用 页

参赛队伍的参赛队号：（请各个参赛队提前填写好）：

4321

竞赛统一编号（由竞赛组委会送至评委团前编号）：

竞赛评阅编号（由竞赛评委团评阅前进行编号）：

The LaTeX Template for tzmcm

Abstract: tzmcmthesis 是 `https://www.latexstudio.net` 为第八届“认证杯”数学中国数学建模国际赛 `http://www.tzmcm.cn` 编写的 LaTeX 模板, 旨在让大家专注于论文的内容写作, 而不用花费过多精力在格式的定制和调整上. 需要注意, 使用者需要有一定的 LaTeX 的使用经验, 至少要会使用常用宏包的一些功能, 比如参考文献，数学公式，图片使用，列表环境等等. 模板已经添加了常用的宏包, 无需用户再额外添加.

关注我们的微信公众号, 获取免费电子书和免费视频:

Keywords: Keywords1 Keywords2 Keywords3

Contents

1. Introduction **5**
1 . 1 5
1 . 2 5
1 . 3 5
2. The Description of the Problem **5**
2 . 1 How do we approximate the whole course of ? 5
2 . 2 How do we define the optimal configuration? 5
2 . 3 The local optimization and the overall optimization 5
2 . 4 The differences in weights and sizes of 5
2 . 5 What if there is no data available? 5
3. Models **5**
3 . 1 Basic Model 5
3 . 1 . 1 *Terms, Definitions and Symbols* 5
3 . 1 . 2 *Assumptions* 5
3 . 1 . 3 *The Foundation of Model* 5
3 . 1 . 4 *Solution and Result* 6
3 . 1 . 5 *Analysis of the Result* 6
3 . 1 . 6 *Strength and Weakness* 6
3 . 2 Improved Model 6
3 . 2 . 1 *Extra Symbols* 6
3 . 2 . 2 *Additional Assumptions* 7
3 . 2 . 3 *The Foundation of Model* 7
3 . 2 . 4 *Solution and Result* 7
3 . 2 . 5 *Analysis of the Result* 7
3 . 2 . 6 *Strength and Weakness* 7
4. Conclusions **7**
4 . 1 Conclusions of the problem 7
4 . 2 Methods used in our models 8
4 . 3 Applications of our models 8
5. Future Work **8**
5 . 1 Another model 8
5 . 1 . 1 *The limitations of queuing theory* 8
5 . 1 . 2 8
5 . 1 . 3 8
5 . 1 . 4 8
5 . 2 Another layout of 9
5 . 3 The newly- adopted charging methods 9
6. References **9**
7. Appendix **10**

I. Introduction

In order to indicate the origin of problems, the following background is worth mentioning.

1.1

1.2

1.3

II. The Description of the Problem

2.1 How do we approximate the whole course of ?

-
-
-

2.2 How do we define the optimal configuration?

1) From the perspective of :
2) From the perspective of the :
3) Compromise:

2.3 The local optimization and the overall optimization

-
-
- Virtually:

2.4 The differences in weights and sizes of

2.5 What if there is no data available?

III. Models

3.1 Basic Model

3.1.1 *Terms, Definitions and Symbols*

The signs and definitions are mostly generated from queuing theory.

3.1.2 *Assumptions*

3.1.3 *The Foundation of Model*

1) The utility function

- The cost of :

- The loss of :
- The weight of each aspect:
- Compromise:

2) The integer programming According to theory, we can calculate the statistical properties as follows.

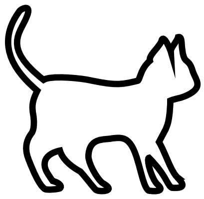

Figure 1 A cat

3) The overall optimization and the local optimization

- The overall optimization:
- The local optimization:
- The optimal number of :

3.1.4 *Solution and Result*

1) The solution of the integer programming: 2) Results:

3.1.5 *Analysis of the Result*

- Local optimization and overall optimization:
- Sensitivity: The result is quite sensitive to the change of the three parameters
-
- Trend:
- Comparison:

3.1.6 *Strength and Weakness*

- Strength: In despite of this, the model has proved that . Moreover, we have drawn some useful conclusions about . The model is fit for, such as
- Weakness: This model just applies to . As we have stated, . That’s just what we should do in the improved model.

3.2 Improved Model

3.2.1 *Extra Symbols*

Signs and definitions indicated above are still valid. Here are some extra signs and definitions.

-
-
-

-

3.2.2 *Additional Assumptions*

-
-
- Assumptions concerning the process are the same as the Basic Model.

3.2.3 *The Foundation of Model*

1) How do we determine the optimal number? As we have concluded from the Basic Model,

3.2.4 *Solution and Result*

1) Simulation algorithm

Based on the analysis above, we design our simulation arithmetic as follows.

- Step1:
- Step2:
- Step3:
- Step4:
- Step5:
- Step6:
- Step7:
- Step8:
- Step9:

2) Flow chart The figure below is the flow chart of the simulation.

3) Solution

3.2.5 *Analysis of the Result*

3.2.6 *Strength and Weakness*

Strength: The Improved Model aims to make up for the neglect of . The result seems to declare that this model is more reasonable than the Basic Model and much more effective than the existing design.

Weakness: Thus the model is still an approximate on a large scale. This has doomed to limit the applications of it.

IV. Conclusions

4.1 Conclusions of the problem

-
-
-
-

4.2 Methods used in our models

-
-
-
-

4.3 Applications of our models

-
-
-
-

V. Future Work

5.1 Another model

5.1.1 *The limitations of queuing theory*

5.1.2

5.1.3

5.1.4

1)

-
-
-
-

2)

-
-
-
-

3)

-
-
-
-

4)

5.2 Another layout of

5.3 The newly- adopted charging methods

VI. References

[1] https://www.latexstudio.net

[2] https://wenda.latexstudio.net

[3] https://github.com/latexstudio/CUMCMThesis

VII. Appendix

Listing 1: The matlab Source code of Algorithm

```
kk=2;[mdd,ndd]=size(dd);
while ~isempty(V)
[tmpd,j]=min(W(i,V));tmpj=V(j);
for k=2:ndd
[tmp1,jj]=min(dd(1,k)+W(dd(2,k),V));
tmp2=V(jj);tt(k-1,:)=[tmp1,tmp2,jj];
end
tmp=[tmpd,tmpj,j;tt];[tmp3,tmp4]=min(tmp(:,1));
if tmp3==tmpd, ss(1:2,kk)=[i;tmp(tmp4,2)];
else,tmp5=find(ss(:,tmp4)~=0);tmp6=length(tmp5);
if dd(2,tmp4)==ss(tmp6,tmp4)
ss(1:tmp6+1,kk)=[ss(tmp5,tmp4);tmp(tmp4,2)];
else, ss(1:3,kk)=[i;dd(2,tmp4);tmp(tmp4,2)];
end;end
dd=[dd,[tmp3;tmp(tmp4,2)]];V(tmp(tmp4,3))=[];
[mdd,ndd]=size(dd);kk=kk+1;
end; S=ss; D=dd(1,:);
```

Listing 2: The lingo source code

```
kk=2;
[mdd,ndd]=size(dd);
while ~isempty(V)
    [tmpd,j]=min(W(i,V));tmpj=V(j);
for k=2:ndd
    [tmp1,jj]=min(dd(1,k)+W(dd(2,k),V));
    tmp2=V(jj);tt(k-1,:)=[tmp1,tmp2,jj];
end
    tmp=[tmpd,tmpj,j;tt];[tmp3,tmp4]=min(tmp(:,1));
if tmp3==tmpd, ss(1:2,kk)=[i;tmp(tmp4,2)];
else,tmp5=find(ss(:,tmp4)~=0);tmp6=length(tmp5);
if dd(2,tmp4)==ss(tmp6,tmp4)
    ss(1:tmp6+1,kk)=[ss(tmp5,tmp4);tmp(tmp4,2)];
else, ss(1:3,kk)=[i;dd(2,tmp4);tmp(tmp4,2)];
end;
end
    dd=[dd,[tmp3;tmp(tmp4,2)]];V(tmp(tmp4,3))=[];
    [mdd,ndd]=size(dd);
    kk=kk+1;
end;
S=ss;
D=dd(1,:);
```