**一个多节点****声纳系统中同步时钟机制的可靠性评估和系统优化问题**

519030910115 李春一

**摘要**

在主从通信系统中，有时会发生系统故障，导致系统的平均寿命/可靠性有限。

节点数量对系统的平均寿命/可靠性起决定性作用，节点过少会导致工作的主从节点数量不足，节点过多可能由于一个节点的故障导致全局崩溃。因此，若想让系统的正常工作时间尽可能长，需要选取合适的节点个数。

针对不同种类的故障及其影响，本文建立了数学模型，将切换器、节点、系统的工作状态视为马尔科夫过程。通过数值模拟方法（包括定步长与变步长两种），验证了不同节点数目下，系统的平均寿命/可靠性。

在本文声纳系统的物理条件下，20个节点平均寿命最长、15个节点可靠性最高。同时，本文分析了A，B两种节点各自的重要性，证明了其中一个切换器的性能进行提升，都会增加整体的平均寿命/可靠性，从而为系统中切换器、节点的设计指明了方向。

**关键词：**系统可靠性 主从通信系统 数值模拟方法 马尔科夫链

目录

[1． 概述 1](#_Toc73036067)

[1.1. 引言 1](#_Toc73036068)

[1.2. 寿命物理模型 1](#_Toc73036069)

[1.2.1. 切换器对节点 1](#_Toc73036070)

[1.2.2. 节点对系统 2](#_Toc73036071)

[1.3. 优化目标 2](#_Toc73036072)

[2. 算法选择 3](#_Toc73036073)

[2.1. 定步长 3](#_Toc73036074)

[2.2. 变步长 3](#_Toc73036075)

[2.3. 算法合理性说明 4](#_Toc73036076)

[3. 最大可靠性 4](#_Toc73036077)

[4. 最大平均工作寿命 5](#_Toc73036078)

[5. 拓展部分 5](#_Toc73036079)

[5.1. 概述 5](#_Toc73036080)

[5.2. 元件角度：提升切换器性能 5](#_Toc73036081)

[5.3. 节点角度：改变节点内部构造 6](#_Toc73036082)

[5.4. 系统角度：提升最大限定寿命 7](#_Toc73036083)

[6. 总结 8](#_Toc73036084)

[6.1. 当下的优缺点、优化方向 8](#_Toc73036085)

[6.2. 结论 8](#_Toc73036093)

# 概述

## 引言

在多节点声纳监听系统中，会发生切换器故障，导致系统无法正常工作。然而，水下的雷达系统不便于修复，因此，设法延长系统的寿命至关重要。

声纳系统由以下三级组成：

* 切换器：分为，两种
* 节点：共个，每个节点包含，两种切换器
* 系统：个节点共同组成完整的声纳系统

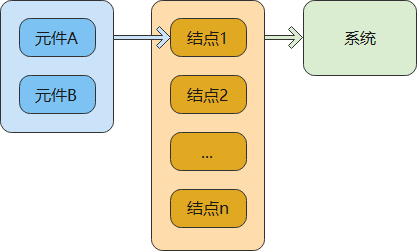


图 1系统组成示意图

## 寿命物理模型

### 切换器对节点

|  |  |  |  |
| --- | --- | --- | --- |
| 切换器 状态 | 切换器 状态 | 节点状态 | 别名 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

表 1节点状态对应表[1]

以上为切换器与节点状态的对应表，切换器A，B分别有3，2种异常状态，这种工作状态的组合，会产生6种不同的节点状态，这些状态的具体情况如下：

* ：节点性能完好
* ：只能作为从节点
* ：或者作为主节点，或者作为不阻塞总线的失效节点
* ：只能作为主节点，否则就会阻塞总线
* ：成为不阻塞总线的失效节点
* ：节点总是阻塞总线

### 节点对系统

本题中，只有个节点工作（包含1个主节点），且总线不被阻塞时，系统正常工作。

基于1.2.1中6种不同的节点状态，可以将时刻六种节点的个数记为，可知节点与系统工作状态的关系。[2]

系统在以下任意情况下不能工作：

* C1：任一节点处于，即
* C2：有两个或以上节点处于，即
* C3：无法找到适合工作于主模式的节点，即
* C4：能构成有效主从系统的节点总数少于个，

系统满足在时能工作：

* C5：无节点处于，即
* C6：有且仅有一个节点处于，即且
* C7：无节点处于，至少一个节点处于 且该节点担当主节点，其余可工作于 从模式的节点数不少于，即的前提下满足 且，或满足且且

系统在同时满足以下情况时有概率能工作：

* C8：无节点处于，即
* C9：处于的节点大于等于 1，且处于 和 的节点总数恰好为，另有若干节点处于，即且且

由于主节点随机选取，

## 优化目标

* 系统可靠性

若系统在内均可靠工作，则判定系统为可靠。

令系统的工作寿命尽量大于，即进行多次模拟，使得寿命超过的次数尽可能多（概率尽可能大）。

* 系统平均工作寿命

令系统的工作寿命尽可能大，即进行多次模拟，使得平均寿命尽可能大。

# 算法选择

## 定步长

按照以下步骤设计算法：

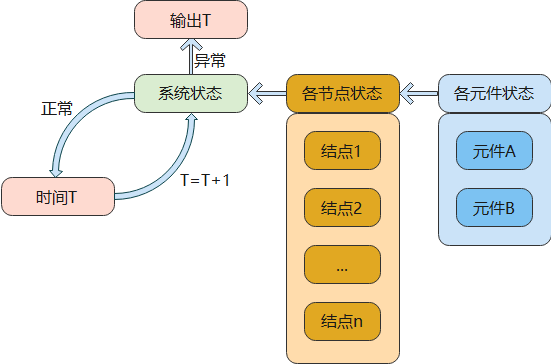


图 2定步长算法示意图

设定步长为，每步改变各元件的状态，改变概率，由此得到各节点的状态，最后模拟出系统的状态。

* 若系统正常，则将寿命增加
* 若系统异常，或总时间大于，输出寿命

此方法可以得出正确结果，但耗时较长。在20个节点的情况下，每尝试100个样本，需要运行。由于使用数值模拟方法，样本数至少为，因此一次计算会耗时接近，因此需要探索速度更快的方法。

## 变步长

按照以下步骤设计算法：

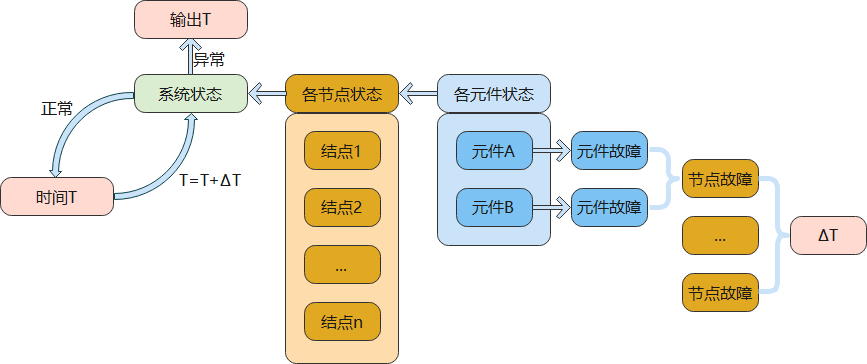


图 3变步长算法示意图

在个节点中，依次模拟个切换器，的状态，得到各个切换器发生故障的时间

，，将最短的时间记为，从而得到最先发生故障的切换器。由此得到各节点的状态，最后模拟出系统的状态。

* 若系统正常，则将寿命增加
* 若系统异常，或总时间大于，输出寿命

此方法可以得出正确结果，且耗时较短。在个节点的情况下，一次计算耗时约。

因此，本文采用变步长算法，设定仿真颗粒度为。

## 算法合理性说明

2.2中的算法可以规避以下两问题：

* 系统失效又复活

当系统异常时，算法会直接跳出循环，时间不会继续增加，即系统不会“复活”。

* 系统永生不死

当寿命大于时，算法自动跳出循环，输出，故系统不存在永生不死的问题。

因此，本文的算法具有合理性。

# 最大可靠性

对个节点进行仿真，得到结果如下：

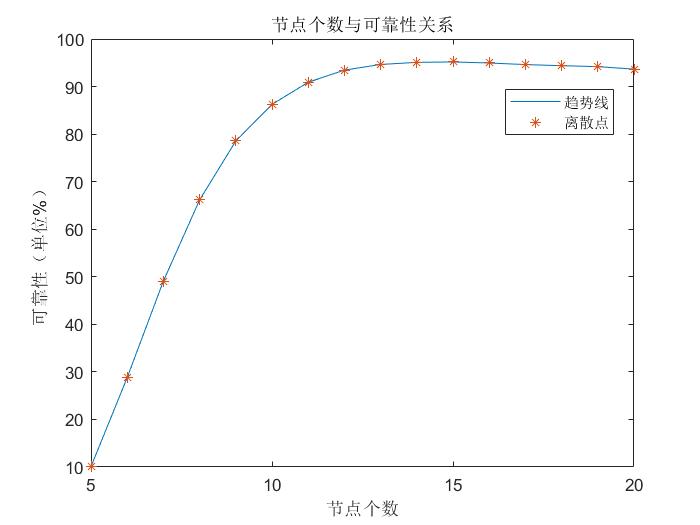


图 4最大可靠性图

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 节点个数 |  |  |  |  |  |
| 可靠性 |  |  |  |  |  |

表 2最大可靠性图

可以看出，节点个数较少时，系统容易发生“工作节点不足”的情况，导致寿命小于；节点个数较多时，总体故障概率增大，更有可能在前发生“总线阻塞”情况。因此，可靠性在节点个数适中时最大。

节点个数：

最大可靠性：

# 最大平均工作寿命

对个节点进行仿真，得到结果如下：

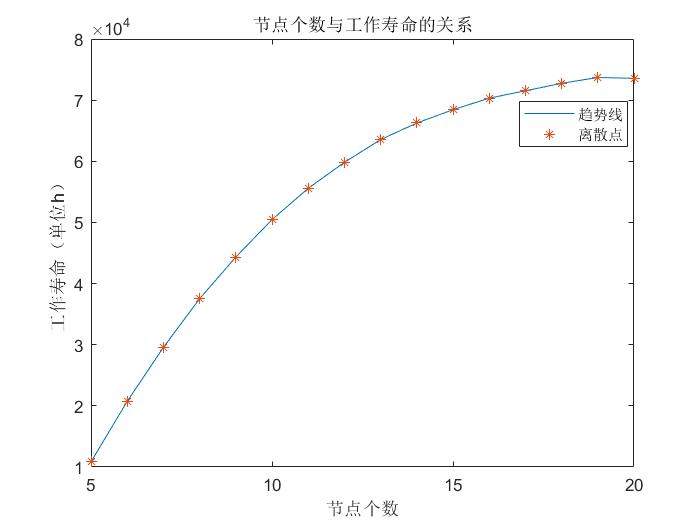


图 5最大工作寿命图

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 节点个数 |  |  |  |  |  |
| 寿命（） |  |  |  |  |  |

表 3最大工作寿命表

可以看出，工作寿命随节点个数增多而增大，增速渐渐放缓，一方面是因为最大寿命的限制，另一方面则是节点个数增多导致“总线阻塞”情况更容易发生。

节点个数：

最大平均工作寿命：

# 拓展部分

## 概述

本文希望通过不同方法，提升系统性能，包括可靠性与最大寿命这两方面。

对此，本文根据元件、节点、系统的“三步走”策略，提出了提升切换器性能，改变节点内部构造，增加最大限定寿命这三种方案。

## 元件角度：提升切换器性能

每个节点由切换器A，B构成，如果能减小切换器的故障概率，即可减小每一个节点的故障概率，从而增加整个系统的预期寿命。

切换器A，B发生故障的情况与有关。因此，将的值调整为原来的，倍，再调用函数，可以提升整体的预期寿命与可靠性。在此，默认节点个数为，以平均寿命的提升为例，以下热力图展示了这一方法的效果：

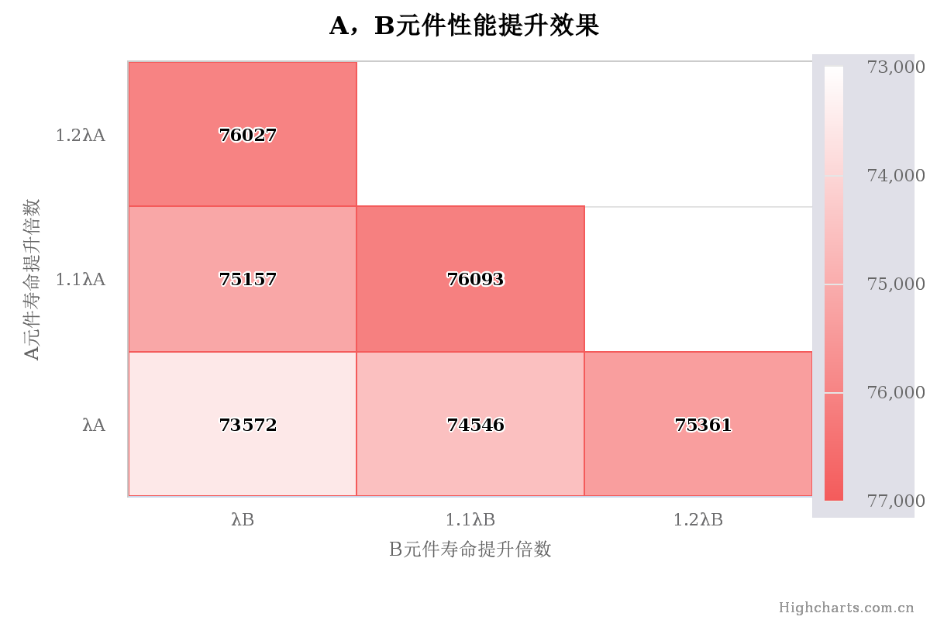


图 6切换器寿命与总体寿命的关系

可以看出，提高可以提高单个切换器的性能，从而提升整体性能，其中提升A切换器性能的效果更为出色。同时，实验结果中的说明，与其专注于提升单个切换器性能，不如同时关注两个切换器，将其性能分别进行适度的提升。

## 节点角度：改变节点内部构造

初始的节点设计中，包含一个切换器和一个切换器，根据[3]中的分析可知，“总线阻塞”对系统寿命影响大，它主要跟切换器故障有关。因此，额外设置一个切换器，通过两个串联的方式，可以有效地减少总线阻塞。

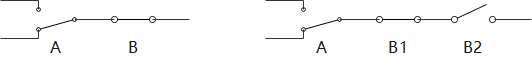


图 7传统节点构造与新节点构造

可以看出，当传统节点的掷刀无法与触点脱离时，若也存在故障，容易发生总线阻塞。然而，新接线方式中，即使处于常闭合状态，的工作状态不受任何影响，因此发生总线阻塞的概率大大降低。

将与视为一个统一的元件，其工作状态的组合会决定整体的状态：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 正常工作 | 常闭合状态 | 常断开状态 |
| 正常工作 | | 正常工作 | 正常工作 | 常断开状态 |
| 常闭合状态 | | 正常工作 | 常闭合状态 | 常断开状态 |
| 常断开状态 | | 常断开状态 | 常断开状态 | 常断开状态 |

表 4整体工作状态与，的关系

同时使用新旧节点构造，对比个节点状态下的工作寿命与可靠性：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 节点数 | 旧寿命（） | 新寿命（） | 提升比例 | 旧可靠性 | 新可靠性 | 提升比例 |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

表 5新旧节点构造效果对比

可以看出，改变节点内部构造的作用在于“防止系统寿命过短”，即防止总线阻塞的情况发生。因此，系统寿命有了大幅度的提升。值得一提的是，由于总线阻塞的情况发生概率大大减少，系统的寿命通常大于，因此提升可靠性效果显著，可以获得接近的可靠性。

## 系统角度：提升最大限定寿命

在进行了节点情况的模拟后，本文发现有系统近一半，其失效不是因为内部故障导致，而是因为寿命触及了最大值，强制失效。

因此，本文希望通过提升最大限定寿命，减少系统寿命触及最大值的概率，从而提升系统的平均寿命。在接下来的模拟中，保持节点个数为，每次将最大限定寿命提升，直到系统寿命触及最大值的概率小于。

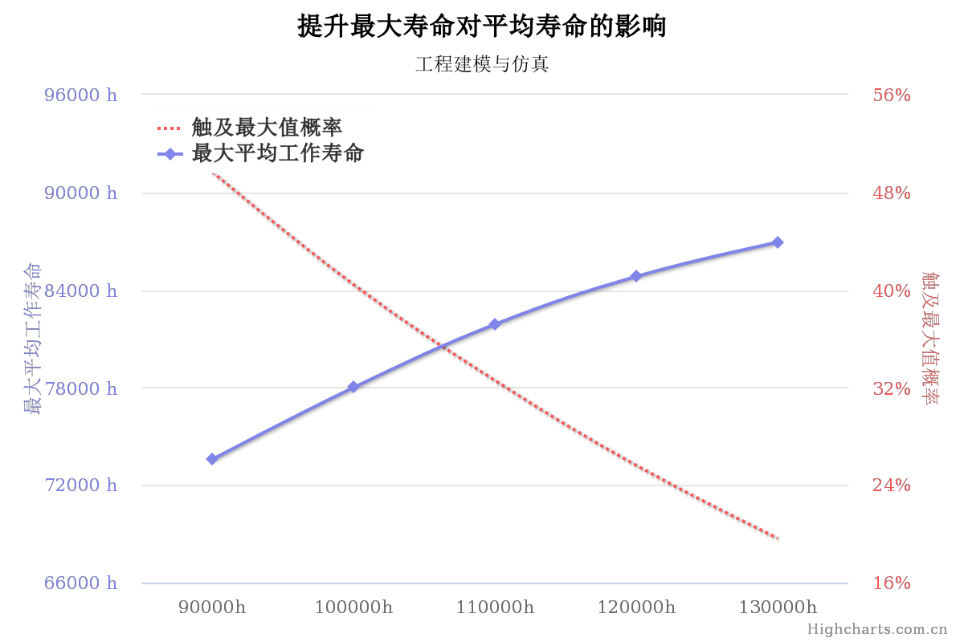


图 8提升最大寿命对平均寿命的影响

可以看出，随着最大限定寿命的增加，系统触及最大值的概率减小，因此系统的平均寿命增大。然而，当最大限定寿命足够大时，触及最大值的概率减小幅度不会很大，因此该方法在前期效果明显，后期对平均寿命的提升不足。因此，需要了解整体的寿命分布情况，才可以选择合适的最大限定寿命，尽量减少强制失效。

依然以节点个数为例，以下为系统寿命的分布图（在原始设置下，系统寿命），可以看出，系统的寿命曲线类似于高斯分布。因此，采用取点法，分别强制丢弃的系统，观察寿命提升效果：

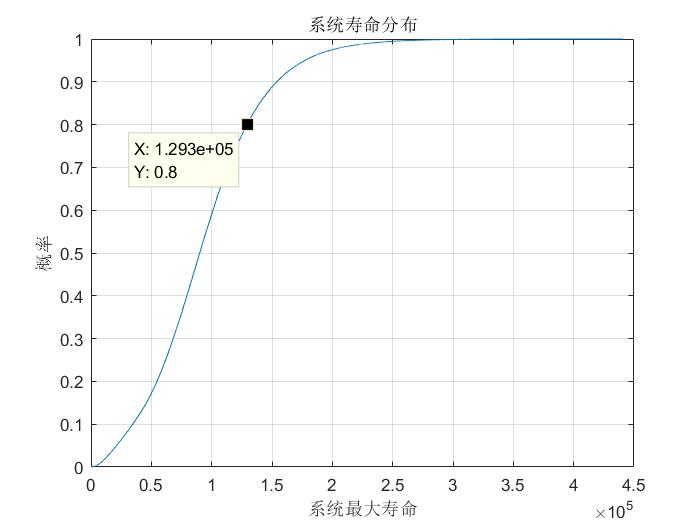
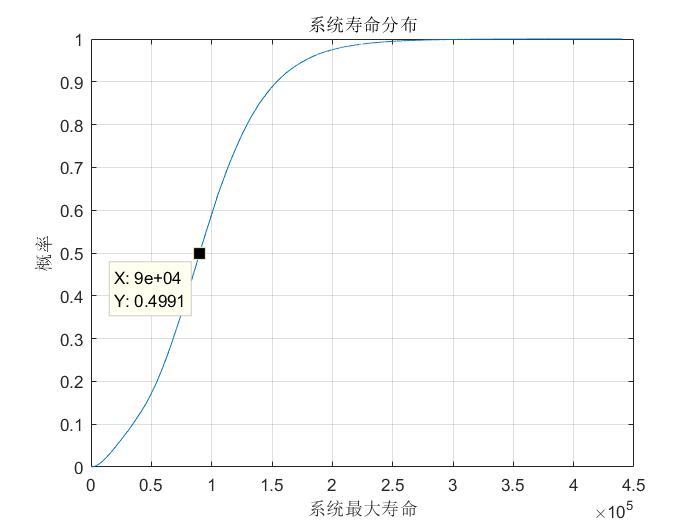


图 9系统寿命分布

|  |  |  |  |
| --- | --- | --- | --- |
| 强制失效 | 最大限定寿命（） | 系统平均寿命（） | 提升比例 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

表 6不同丢弃比例对平均寿命的影响

结论显示，最大限定寿命越大，强制失效比例越小，系统的平均寿命越大，符合预期。

# 总结

## 当下的优缺点、优化方向

优点：

* 选择了变步长方案，并证明了其时间远远优于定步长方法
* 规避了算法中“复活”与“永生”的漏洞
* 得出了稳定性与平均寿命最大情况下的结点个数，并进行了分析
* 提出元件-节点-系统三级方案优化系统寿命，并采用多种可视化方式展示优化结果

不足：

* 系统角度的优化方案只能提升寿命，无法像元件、节点角度方案一样提升可靠性
* 运行时间仍有优化空间，运行计时中显示函数占用了运行时间的一半
* 不同方案下的结点数说明

优化方向：

* 当，变化，或最大限定寿命升高时，最优节点数不一定为个，应在三种优化方案中，对节点数为的情况依次尝试。（因算力问题只以节点为例）



## 结论

通过本文研究发现，在题目条件下，节点数为时，可靠性最大；节点数为时，寿命最大。

从元件维度，可以通过提升，各自的性能来提高整体寿命与可靠性，效果较好；

从节点维度，可以增加一个元件提升寿命与可靠性，其中可靠性增至近；

从系统维度，可以提升最大限定寿命从而大幅提升平均寿命，但对可靠性无作用。

**附录**

**参考文献**

[1]上海交通大学 电子工程系 工程问题建模和仿真讲座15

[2]上海交通大学 电子工程系 案例2系统可用性数值的理论求解方法介绍

[3]上海交通大学 电子工程系 工程问题建模与仿真讲座16

**变步长代码**

Nsample=100000;

k=5; %工作结点数

pA=[0.26,0.26,0.48];

pB=[0.35,0.65];

pointOutput=zeros(1,n);

resOutput=zeros(1,n);

for numOfPoint=5:20 %总结点数

life=zeros(1,Nsample);

for t=1:Nsample

stateSystem=0;

stateComp=zeros(1,numOfPoint);

stateItemA=zeros(1,numOfPoint);

stateItemB=zeros(1,numOfPoint);

while stateSystem==0

[stateItemA,stateItemB,T]=changeStateItemT(stateItemA,stateItemB,pA,pB,numOfPoint);

stateComp=changeStateComp(stateItemA,stateItemB);

stateSystem=judgeSystem(stateComp);

life(t)=life(t)+T;

if life(t)>=90000

life(t)=90000;

break

end

end

%if mod(t,100)==0

% fprintf('%d sample,meanLife=%f \n',t,sum(life)/t)

%end

end

pointOutput(numOfPoint)=mean(life);

resOutput(numOfPoint)=sum((life-25000)>=0)\*100/Nsample;

fprintf('numOfPoint=%d, meanLife=%f \n',numOfPoint,pointOutput(numOfPoint))

fprintf('numOfPoint=%d, meanRes=%f %% \n',numOfPoint,resOutput(numOfPoint))

end

function [stateItemA,stateItemB,T]=changeStateItemT(stateItemA,stateItemB,pA,pB,n)

% 改状态

lambdaA=37000;

%lambdaA=lambdaA\*1.1;

lambdaB=480000;

%lambdaB=lambdaB\*1.1;

T1=exprnd(lambdaA,n,1);

T2=exprnd(lambdaB,n,1);

T1min=min(T1);

T2min=min(T2);

T1Index=find(T1==T1min);

T2Index=find(T2==T2min);

if T1min<=T2min && stateItemA(T1Index)==0

path=rand(1);

if path<pA(1)

stateItemA(T1Index)=1;

elseif pA(1)<=path<(pA(1)+pA(2))

stateItemA(T1Index)=2;

else

stateItemA(T1Index)=3;

end

end

if T1min>=T2min && stateItemB(T2Index)==0

path=rand(1);

if path<pB(1)

stateItemB(T2Index)=1;

else

stateItemB(T2Index)=2;

end

end

T=min(T1min,T2min);

end

function stateComp=changeStateComp(stateItemA,stateItemB)

%AB工作状态决定整体

changeMat=[0 3 1;1 5 1;2 3 4;4 4 4];

p=length(changeMat);

stateComp=changeMat(stateItemA+stateItemB\*p+1);

end