主从多机通信系统可靠性问题的建模

组号：026 李澄曜 5140219288 林特 5140219432

摘要：为了定量的描述一个产品在特定时间，特定条件下不失效地发挥其功能的概率，我们引入了可靠性的概念并将其作为评判该产品实用性的参量。本文通过分析一个主从式多节点声呐系统中同步时钟机制，以其实际系统为研究对象进行建模，模拟系统中各器件的工作状态、使用寿命以仿真其在实际系统中的状态。通过理论分析、算法研究及仿真实现，得出不同节点数下系统平均寿命及在给定时间内的可靠性。

关键字：主从多机，可靠性，使用寿命，仿真模拟

Modeling of Reliability of Master-slave Communication System

Abstract: We use reliability to describe the probability of a product working well in certain time and conditions and judge its practicality. By means of analyzing a synchronous clock mechanism of a master-slave nodes sonar system, this paper makes mathematical models on the practical application and stimulates the life and working conditions of electronical devices. Based on the study of theorem and algorism, we can get the simulative life and reliability of the systems with different numbers of nodes.

Key words: Master-slave Communication System, reliability, analogue simulation

1 引言

随着电子器件的自动化越来越普及，可靠性作为一个系统能否在无人条件下，进行正常工作的重要指标，其地位愈发重要。所以对可靠性进行定量评价有诸多现实意义，如指导不同硬件电路设计方案的优选，指导软件可靠性优化方案的设计，指导优化网络物理拓扑，指导容错冗余措施的优化部署等。而模拟仿真则是定量研究可靠性的一种有效手段，本文对一个多节点声纳系统中同步时钟机制可靠性的各类参数进行了仿真分析，同时解出了最优状态时的系统的节点个数及最长平均失效时间。

2 物理模型及其参数

2.1 模型简述

如图 2.1 所示，某分布式部署的声纳系统共有 n 个独立节点构成。各节点内部均是物理同构的。各节点必须保持严格的时钟信号同步才能有效协同工作，使系统发挥作用。所有

节点经由时钟信号总线连接，由其中一个节点担当主节点，它的时钟电路工作于主模式，向总线输出时钟信号；其余节点均应担当从节点，节点内部时钟电路工作于从模式，仅从总线获取信号，不向总线输出信号。

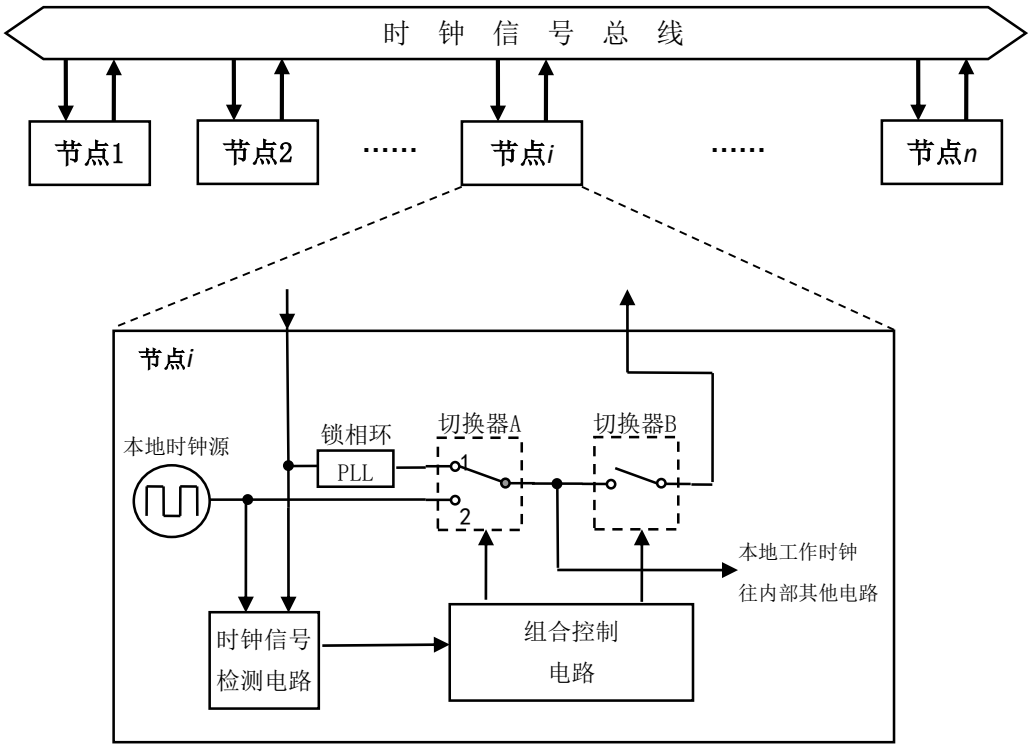


图 2-1 一个多节点声纳系统中的时钟同步机制示意图

2.2 理论假设与模型参数

2.2.1 模型中的元件

为降低运算复杂度，我们将切换器 A 和切换器 B 视作不可靠元件，而将系统中的其余元件均视作不失效的可靠元件，这些元件的失效风险已被等效地折算计入不可靠元件的失效风险中。

切换器可视作一种多状态元件，且彼此特性统计独立。由各元件组成的节点是构成系统的部件，显然其性能表现也是多状态的。声纳系统整体可看作一个多状态系统。

2.2.2 不可靠元件的使用寿命和各种故障的发生概率

切换器元件使用寿命的概率密度分布都遵从负指数分布。

(1) 切换器 A 的使用寿命T_A的概率密度分布和各种故障概率

$$f_{T_A}(\tau)=\lambda_A e^{-\lambda_A \tau} \qquad \text{其中 } 1/\lambda_A = 9.2\times 10^4 h$$

故障 A1：切换器 A 不能正常受控，掷刀无法与触点 1 脱离。

条件概率：$P_{EA1}=\Pr(\text{A1 occers} \mid \text{A is failed})=0.19$

故障 A2：切换器 A 不能正常受控，掷刀无法与触点 2 脱离。

条件概率：$P_{EA2}=\Pr(\text{A2 occurs} \mid \text{A is failed})=0.25$

故障 A3：切换器 A 不能正常受控，掷刀无法与任何一个触点接合。

条件概率： $P_{EA3} = \Pr(\text{A3 occurs} \mid \text{A is failed}) = 0.56$

(2) 切换器 B 的使用寿命 T_B 的概率密度分布

$f_{T_B}(\tau) = \lambda_B e^{-\lambda_B \tau}$ 其中 $1/\lambda_B = 3.9 \times 10^5 h$

故障 B1：切换器 B 不能正常受控，掷刀无法与触点脱离。

条件概率： $P_{EB1} = \Pr(\text{B1 occurs} \mid \text{B is failed}) = 0.65$

故障 B2：切换器 B 不能正常受控，掷刀无法与触点接合。

条件概率： $P_{EB2} = \Pr(\text{B2 occurs} \mid \text{B is failed}) = 0.35$

2.2.3 元件的状态

切换器 A 是一个 4 状态元件，g_{A0} 表示其处于正常工作状态，而 g_{A1}、g_{A2}、g_{A3} 分别表示处于故障 A1、A2、A3 状态。任一节点 *i* 中的切换器 A 状态

$G_{A_i}(t) \in \{g_{A0}, g_{A1}, g_{A2}, g_{A3}\} \quad i = 1,2,3,\ldots n$

切换器 B 是一个 3 状态元件，其他同理。

2.2.4 节点的状态

g_{N0} 表示节点性能完好，为直观起见，定义别名 g_{PF}（意为 perfectly functioning）；g_{N1} 表示只能作为从节点，别名 g_{SO}（slave only）；g_{N2} 表示或者作为主节点，或者作为不阻塞总线的失效节点，别名 g_{DM}（disable / master）；g_{N3} 表示只能作为主节点，否则就会阻塞总线，别名 g_{MO}（master only）；g_{N4} 表示成为不阻塞总线的失效节点，别名 g_{DN}（disable node）；g_{N5} 表示节点总是阻塞总线，别名 g_{FB}（failed bus）；

现定义 $Q_{N0}(t)$ 表示时刻 *t* 恰好处于 g_{N0} 状态的节点总数

$$Q_{N0}(t) = \sum_{i=1}^{n} 1(G_{N_i}(t) = g_{N0})$$

则有

条件	含义
C1	$Q_{FB}(t) \geq 1$
C2	$Q_{MO}(t) \geq 2$
C3	$Q_{PF}(t) + Q_{MO}(t) + Q_{DM}(t) = 0$
C4	$Q_{PF}(t) + Q_{SO}(t) + 1\left(\left(Q_{MO}(t) + Q_{DM}(t)\right) > 0\right) < k$
C5	$Q_{FB}(t) = 0$
C6	$Q_{MO}(t) = 1$且$Q_{PF}(t) + Q_{SO}(t) \geq k - 1$

C7	$Q_{MO}(t)=0$且$Q_{PF}(t)\geq 1$且$Q_{PF}(t)+Q_{SO}(t)\geq k$ 或$Q_{MO}(t)=0$且$Q_{PF}(t)=0$且$Q_{DM}(t)\geq 1$且$Q_{SO}(t)\geq k-1$
C8	$Q_{FB}(t)+Q_{MO}(t)=0$
C9	$Q_{PF}(t)+Q_{SO}(t)=k-1$ 且 $Q_{DM}(t)\geq 1$

表 2-1 可能导致系统失效的条件及其数学表达式

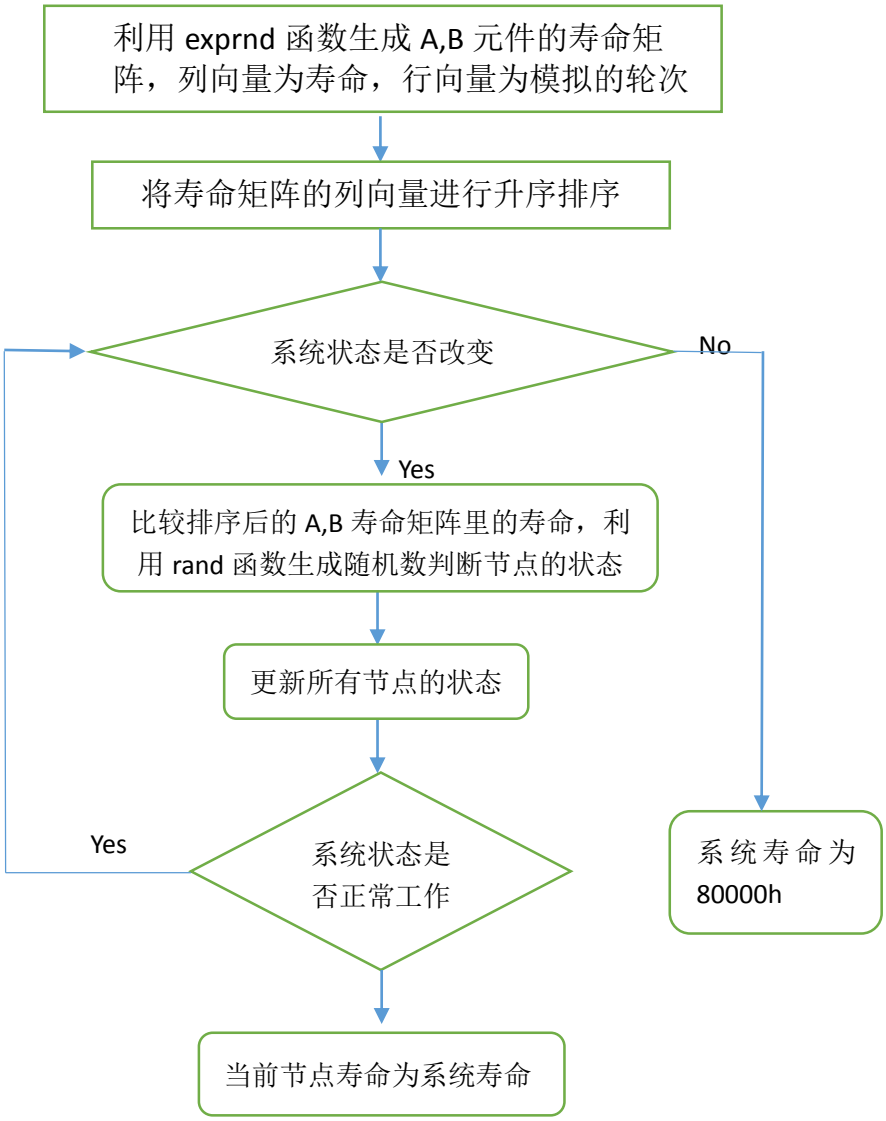
2.2.5 系统的状态

系统状态	状态含义	所需条件
G_{sys1}	系统确定不能有效工作	C1 ∪ C2 ∪ C3 ∪ C4
G_{sys2}	系统确定能有效工作	C5 ∩ (C6 ∪ C7)
G_{sys3}	系统恰能有效工作	C8 ∪ C9且任意 g_{DM} 节点作主节点
G_{sys4}	系统恰不能有效工作	C8 ∪ C9且任意 g_{PF} 节点作主节点

表 2-2 系统状态及其条件关系

3 算法设计与仿真流程图

3.1 仿真流程图



3.2 算法设计

因为定步长的算法虽然思路简单，但复杂度太高，调试与检测都要耗费大量的时间，所以我们考虑以不定步长的思想进行仿真。通过 MATLAB 内置函数 exprnd，系统可以自动生成指数分布的随机数。这些随机数就是系统发生故障的时间点。再通过 rand(1)函数，生成 0-1 之间的随机数，可以模拟节点发生故障的类型。这样，通过 exprnd 和 rand 两个 MATLAB 自带的函数，就可以产生所有仿真过程中需要的随机数了。

当节点数为 m 时，使用 exprnd 函数产生两组参数分别为 lambda A 和 lambda B 的随机数 A，B，每组 m 个。分别代表 A 切换器和 B 切换器状态改变的时间。将 A，B 分别从小到大排列。A 和 B 的第一个元素中的较小值就是系统状态首次发生改变的时间。将此时间记录，若系统不能继续正常工作，该时间就是系统的首次失效时间。若系统仍能继续工作，则再次选择之后的时间节点，判断系统是否能正常工作。

若系统的所有节点都能正常工作或者时间超过最大时间 80000 小时，则算系统寿命为 80000 小时。

在判断系统失效之前将本次模拟的时间存入系统寿命矩阵 t(m,s)，m 为节点数，s 为模拟次数，最后根据寿命矩阵统计模拟过程中的系统可靠性和系统平均工作寿命。

4 实验结果

节点数 n	系统平均寿命 t/h	系统可靠性 η
5	24101	0.3660
6	41411	0.6827
7	51987	0.8345
8	58872	0.8931
9	63151	0.9123
10	65309	0.9139
11	65640	0.8984
12	65615	0.8891

表 4-1 实验结果

其中系统平均寿命 t 表示 10000 次模拟后所得有效寿命的平均值，系统可靠性 η 表示系统寿命超过 25000 小时的概率。从表格中我们可以得出，n=11 时，即系统拥有 11 个节点时系统平均寿命最长；n=10 时系统寿命超过 25000 小时的概率即系统可靠性 η 最大。经过与老师所给参考答案的比对，系统平均寿命的偏差值最大不超过 3.0%，系统可靠性的偏差值最大不超过 3.5%，考虑到每次仿真结果在小范围内的波动，我们认为我们所建立的模型可以正确的描述题设情境。因此我们可以得出结论，在误差允许范围内，节点数 10-11 个的主从多机系统较为稳定。

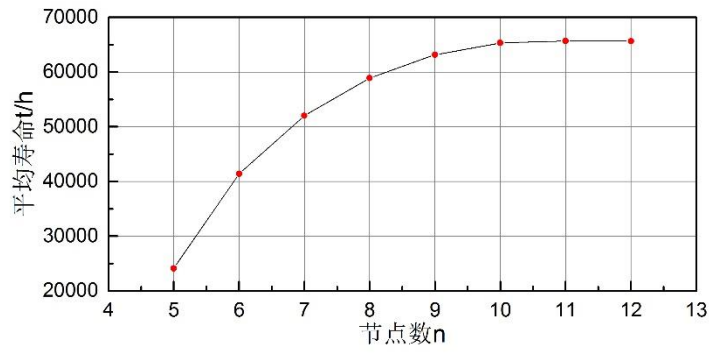


图 4-1 系统平均寿命 t 随节点数 n 的变化

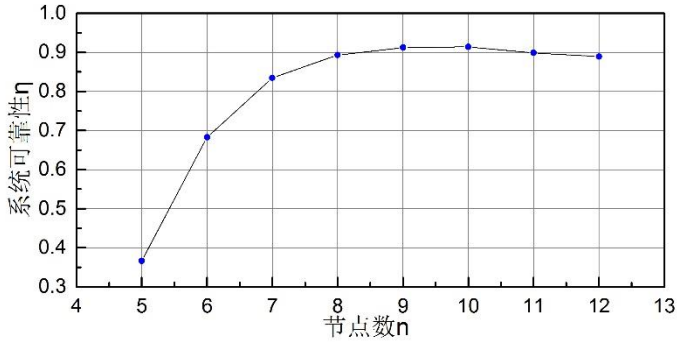


图 4-2 系统可靠性 η 随节点数 n 的变化

从图中我们可以看出，在节点数较少时，任一节点的故障都有较大可能导致整个系统故障，系统寿命和可靠性指标均较差。向系统逐步添加冗余节点后，上述两指标均有较快增长，并在 10-11 个节点数时达到最大。我们试图继续向该系统添加节点（最多达 14 个），观察到平均寿命和可靠性均有小幅下降，但仍维持在较高水平。

5 拓展探究

我们研究主从多机通信系统的目的即是尽可能的提高该系统的寿命及可靠性。上述实验中我们探究了节点数对该系统的影响，而除此之外，A、B 元件自身的性能也是该系统中的重要变量。因此，我们尝试研究了 A、B 两元件的寿命对系统寿命及可靠性的影响。

5.1 元件 A 对系统寿命及可靠性的影响

我们取节点数变化范围为 5-14，保持 B 的寿命维持在3.9×10^5h不变，分别取 1.0 倍、1.2 倍、1.4 倍、1.6 倍、1.8 倍及 2.0 倍 A 寿命进行仿真，观察系统平均寿命和系统可靠性随节点数 n 的变化关系。结果如下：

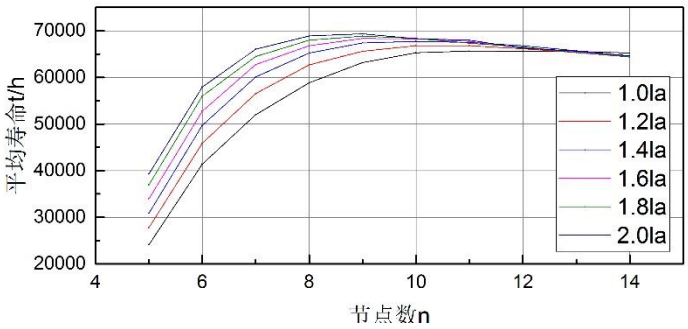


图 5-1 元件 A 寿命增长过程中系统平均寿命 t 随节点数 n 的变化

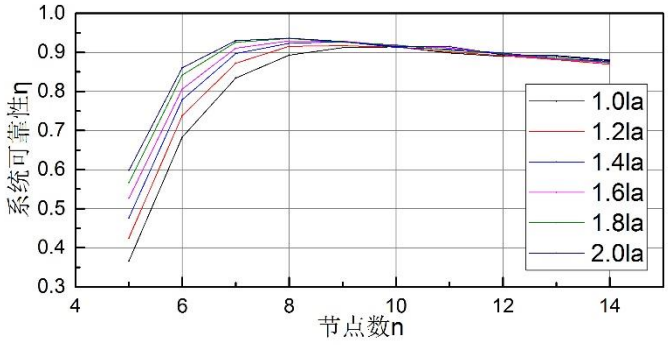


图 5-2 元件 A 寿命增长过程中系统可靠性 η 随节点数 n 的变化

相对 A 原寿命比值 λ	系统平均寿命最长节点数n_1	最长平均寿命提升百分比δ_1	系统可靠性最高节点数n_2	最高可靠性提升百分比δ_2
1.0	11	0	10	0
1.2	10	1.8%	9	0.3%
1.4	10	3.2%	9	1.5%
1.6	9	4.2%	8	1.7%
1.8	9	4.9%	8	2.4%
2.0	9	5.6%	8	2.4%

表 5-1 提高元件 A 性能时系统性能提升情况

从图中我们可以了解到，提高元件 A 的寿命在节点数较少时对系统平均寿命及可靠性的提高更为显著，而节点数较多时性能则稍有下降，甚至弱于节点数相同的原寿命对照组。同时，所有组别可靠性均先于平均寿命达到最大值。

而从所列表中可以得到，取得系统平均寿命最长及可靠性最高的节点数均在减小。即随着元件 A 寿命的提升，使用的节点越少，系统的性能（包括平均寿命及可靠性）更好。同时，随着 A 寿命均匀上升，系统平均寿命的提升速率有所下降。当元件 A 寿命提升至原有 2 倍时，系统平均寿命提升了 5.6%。这也意味着存在一个元件 A 寿命提升的范围，使得其在合理的成本下取得合适的性能提升。而系统可靠性的提升也在逐步上升，但增长已经出现停滞，意味着此时限制系统可靠性的因素更多的取决于元件 B 的寿命以及其他条件。

5.2 元件 B 对系统寿命及可靠性的影响

同样地，我们取节点数变化范围为 5-14，保持 A 的寿命维持在9.2×10^4h不变，分别取 1.0 倍、1.2 倍、1.4 倍、1.6 倍、1.8 倍及 2.0 倍 B 寿命进行仿真，结果如下：

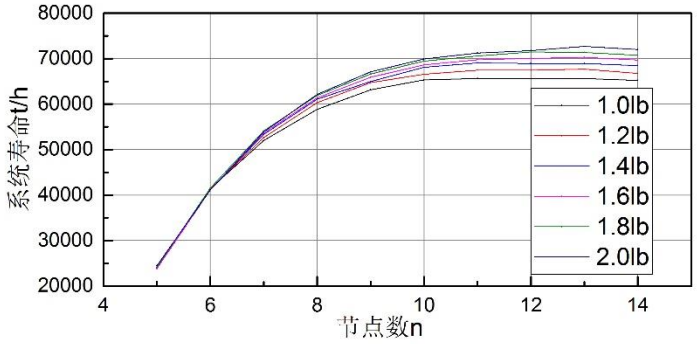


图 5-3 元件 B 寿命增长过程中系统平均寿命 t 随节点数 n 的变化

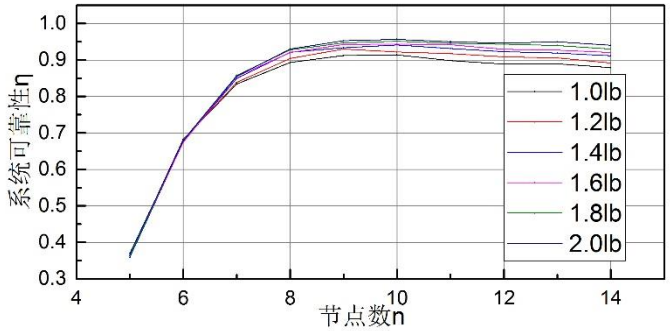


图 5-4 元件 B 寿命增长过程中系统可靠性 η 随节点数 n 的变化

相对 B 原寿命比值 λ	系统平均寿命最长节点数n_3	最长平均寿命提升百分比δ_3	系统可靠性最高节点数n_4	最高可靠性提升百分比δ_4
1.0	11	0	10	0
1.2	13	3.1%	9	1.8%
1.4	11	5.2%	10	3.0%
1.6	13	7.1%	10	3.2%
1.8	12	8.8%	10	4.1%
2.0	13	10.7%	10	4.7%

表 5-2 提高元件 B 性能时系统性能提升情况

有趣的是，相比于元件 A 寿命提高节在点数较少时对系统平均寿命及可靠性的提高更为显著，提高元件 B 的寿命则在节点数较多时发挥了更大的作用。同时，取得系统平均寿命最长的节点数有所上升，意味着随着元件 B 寿命的提升，使用的节点越多，系统的平均寿命越长。而与元件 A 相同的是，平均寿命及可靠性提升的速率均有所下降。

需要注意的是，等比例的提升元件 A 及元件 B 的寿命，元件 B 对系统性能的影响更大一些。我们猜测，元件 B 的模型较 A 更为简单，单纯的提高元件 B 的寿命能更快的改变系统性能，取得更好的效果。

5.3 总结

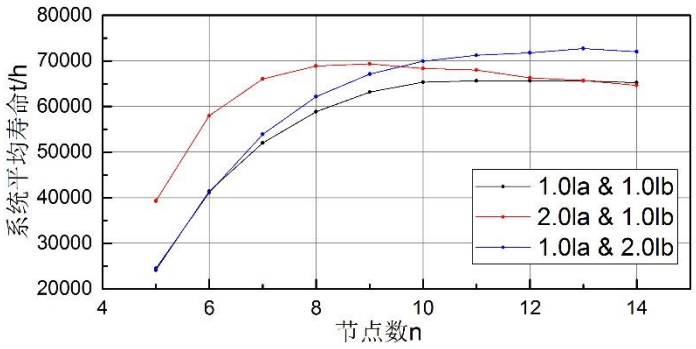


图 5-5 分别提高元件 A、B 寿命对系统寿命的影响

结合上述分析，我们可以看出若只能选择改进一个元件的性能，我们应选择提高元件 B 的寿命。但是相对的，改进 B 的寿命后我们需要更多的节点（意味着更高的成本）才能获得更长的系统寿命以及更高的系统可靠性，而改进 A 的寿命则与之相反。因此，适当的增加元件 A、B 的寿命，可以维持在一个合理的节点数下，获取相对更好的系统性能。

6 参考文献

[1] “主从多机通信系统可靠性建模研究”案例一的要求 V2.61.doc
[2] 指导材料：案例一问题的理论估算方法介绍.doc
[3] 课程设计 讲座 2：案例一.pptx

附录

所使用的 MATLAB 代码

```
la = 92000; %lambda A
lb = 390000; %lambda B
number = 10000; %模拟次数
for m = 5:1:12
    ra = exprnd(la,m,number);
    rb = exprnd(lb,m,number);
    ra = sort(ra); %将随机数从小到大排列
    rb = sort(rb);
    t = 0;
    for s = 1:1:number %开始 number 次模拟
        pa = 1;
        pb = 1;
        ppa = 0; %用来记录系统状态变化
        ppb = 0;
        for k = 1:1:m %初始化
            a(k) = 0; %第 k 个节点 A 的状态
            b(k) = 0; %第 k 个节点 B 的状态
            n(k) = 0; %第 k 个节点的节点状态
        end
            while ppa ~= pa || ppb ~= pb %系统状态变化
                ppa = pa;
                ppb = pb;
                if ra(pa,s) < rb(pb,s) && a(pa) == 0 %切换器 A 先发生故障
                    r = rand(1);
                    t = ra(pa,s);
                if r < 0.19
                    a(pa) = 1;
                elseif r < 0.44
                    a(pa) = 2;
                else
                    a(pa) = 3;
                end %随机 A 的故障类型
                if pa < m
                    pa = pa + 1; %系统状态变化
                end
                elseif b(pb) == 0 %切换器 B 先发生故障
                    r = rand(1);
                    t = rb(pb,s);
                if r < 0.65
                    b(pb) = 1;
                else
```

```
            b(pb) = 2;
        end %随机 B 的故障类型
        if pb < m
            pb = pb + 1; %系统状态变化
        end
    end

    for k = 1:1:m %节点状态更新
        sum = a(k)*10 + b(k);
        if sum == 0
            n(k) = 0;
        elseif sum == 2 || sum == 10 || sum == 12
            n(k) = 1;
        elseif sum == 20
            n(k) = 2;
        elseif sum == 1 || sum == 21
            n(k) = 3;
        elseif sum == 22 || sum == 30 || sum == 31 || sum ==22
            n(k) = 4;
        elseif sum == 11
            n(k) = 5;
        end
    end
    c5 = 0;
    c6 = 0;
    c7 = 0;
    c8c9 = 0; %系统能否正常工作的判断条件
    cnt_n0 = 0; %pf 的个数
    cnt_n1 = 0; %so 的个数
    cnt_n2 = 0; %dm 的个数
    cnt_n3 = 0; %mo 的个数
    cnt_n5 = 0; %fb 的个数
    for k = 1:1:m %计算 n0 到 n5 的个数
        if n(k) == 5
            cnt_n5 = cnt_n5 + 1;
        end
        if n(k) == 3
            cnt_n3 = cnt_n3 + 1;
        end
        if n(k) == 0
            cnt_n0 = cnt_n0 + 1;
        end
        if n(k) == 1
            cnt_n1 = cnt_n1 + 1;
```

```
                end
                if n(k) == 2
                    cnt_n2 = cnt_n2 + 1;
                end
            end

            if cnt_n5 == 0 %判断 C5 是否成立
                c5 = 1;
            end
            if cnt_n3 == 1 && cnt_n0 + cnt_n1 > 2 %判断 C6 是否成立
                c6 = 1;
            end
            if (cnt_n3 == 0 && cnt_n0 > 0 && cnt_n0 + cnt_n1 > 3)
||(cnt_n0 == 0 && cnt_n3 == 0 && cnt_n2 > 0 && cnt_n1 > 2)
                c7 = 1; %判断 C7 是否成立
            end
            if (cnt_n5 + cnt_n3 == 0 && cnt_n0 + cnt_n1 ==3) %判断 C8C9
是否成立
                r = rand(1);
                if r < cnt_n2 / (cnt_n2 + cnt_n0)
                    c8c9 = 1;
                end
            end
            if ~((c5 && (c6 || c7)) || c8c9) %判断系统是否能有效工作
                time(m,s) = t; %记录系统工作的最大时间
            break; %不能继续工作则退出环
        end
    end
end
    count(m) = 0; %系统可靠性
    sumoftime(m) = 0; %系统平均工作寿命
    for s = 1:1:number
        if time(m,s) >= 25000 || time(m,s) == 0
            count(m) = count(m) + 1; %计算能工作 25000h 的系统
        end
        if time(m,s) == 0 || time(m,s) > 80000
            sumoftime(m) = sumoftime(m) + 80000;
        else
            sumoftime(m) = sumoftime(m) + time(m,s);%计算系统总寿命
        end
    end
    count(m) = count(m)/number; %计算可靠性
    sumoftime(m) = sumoftime(m)/number; %计算系统平均工作寿命
end
```