

北京理工大学
2006 年全国大学生数学建模竞赛建模辅导
选修课程作业

姓名	林健
学号	20040200
学院	计算机科学技术学院
班级	01110407

姓名	范舟斌
学号	20040187
学院	计算机科学技术学院
班级	01110407

2006 年 5 月

基于复杂系统理论的计算机病毒传播模型的 计算机仿真分析

摘要

现实生活中，存在大量具有众多状态变量、反馈结构复杂、输入与输出呈现非线性特征的复杂系统。现有的理论不能通过简单的公式或统计方法来精确地研究它们的行为。目前人们对复杂系统的描述和研究，主要是将多学科知识综合，结合定性研究和定量分析方法，对低层次子系统进行宏观集成，对系统较高层次部分微观化处理，通过计算机仿真技术建立系统模型并调试模型参数，同时辅以知识工程技术（如专家系统）等。复杂系统理论作为知识爆炸的时代信息整合的工具，是未来科学发展的必然选择。

本文以计算机病毒传播模型为例，通过设定实时状态参量、状态转移因子，表达计算机病毒传播系统在各个时刻的不同状态和发展趋势。使用计算机编程模拟复杂系统中的诸多因素的相互作用及系统长期演化过程，比较成功地演示了计算机病毒在不同条件下传播的趋势，同时给出了具有针对性的病毒防控方案。计算机仿真方法充分利用了计算机强大的逻辑功能和计算能力，是一种初等的、容易理解的复杂系统分析方法。与传统的微分方程乃至更复杂的数学模型相比，它更注重问题本身的性质，对数学理论的要求相对较低，同时对各类现实问题的适应性和应变性强，是应用领域值得推广的方法。

关键词

复杂系统，计算机病毒，计算机仿真

1 绪论

1.1 复杂系统理论概述

我们生活的世界从宏观角度考虑，可以认为是由各种大大小小的系统构成的。所谓系统，是指一个具有某些特定功能的有机整体。它的各部分相互影响、相互依赖，形成一定的结构与层次。系统从其内子系统的关联关系角度可划分为简单系统与复杂系统。

所谓复杂系统，其最主要特征是系统具有众多的状态变量，反馈结构复杂，输入与输出呈现非线性特征。依照经典的信号与系统分析理论，复杂系统可称为高阶、非线性、时变、多回路系统。常见的社会经济系统、人体生理系统等都是复杂系统。这些系统在结构、功能、行为、演化等方面十分复杂，至今仍有大量问题还不被了解。

目前人们对复杂系统的描述和研究，采取了一些特殊的方法。总体而言，主要是将多学科知识综合，结合定性研究和定量分析方法，对低层次子系统进行宏观集成，对系统较高层次部分微观化处理，通过计算机仿真技术建立系统模型并调试模型参数，同时辅以知识工程技术（如专家系统）等。

1.2 计算机病毒问题概述

依据《中华人民共和国计算机信息系统安全保护条例》，计算机病毒是指编制或者在计算机程序中插入的破坏计算机功能或者毁坏数据，影响计算机使用，并能自我复制的一组计算机指令或者程序代码。

计算机病毒从执行机理方面，一般分为文件型病毒、引导型病毒、宏病毒、木马病毒、脚本病毒、蠕虫病毒等。无论哪种病毒，之所以能被称为“病毒”，是因为它与生物病毒在原理上相似，具有生命系统所固有的特性——繁殖、机体集成、不可预见性等。计算机病毒的生命周期也类似于生物病毒，包括潜伏阶段、传染阶段、触发阶段、发作阶段等。因此研究计算机病毒的方法与研究生物病毒有一定的可比性。病毒技术和反病毒技术这对矛盾在不断的斗争中持续发展，是信息社会安全领域的重要课题。

2 计算机病毒传播模型分析

2.1 问题分析

在信息社会，整个信息网络（包括计算机网络、电信网络、广播电视网络等）构成一个巨大的系统。网络上的每一台设备（如计算机）又是一个相对独立的子系统。计算机中的每一个程序也是一个具有特定功能的更小的子系统。

一般的计算机程序，是由编程人员依据用户的需要而开发，它们接收输入数据，按照一定的业务逻辑处理数据，然后给出合理的输出。无论其外部行为还是内部机理，都可以看做是一个“接收信号——对信号流进行线性处理——输出信号”的简单系统。计算机程序与在用户的控制下，与数据、网络及其它计算机程序等外部系统发生可以预见的联系。

计算机病毒也是一类程序，具有一般程序的共性。但更重要的是病毒具有非法性、隐藏性、潜伏性、可触发性、表现性、破坏性、传播性、针对性、变异性和不可预见性等特征。其中传播性决定了研究病毒行为不可以像研究一般程序那样在单个计算机系统上进行，必须依赖于多个计算机系统，乃至网络系统。可触发性、不可预见性决定了病毒的各类行为构成一个近乎随机的系统（虽然对特定的病毒，其执行和传

染方式是确定的，但由于触发条件的多样性、用户操作的随机性、网络条件的不确定性，可以认为病毒是随机系统)。隐藏性、潜伏性使得我们不能建立一个透明的白箱模型对其进行研究。变异性进一步增加了我们对病毒模型构建的复杂程度。非法性、破坏性、针对性、表现性等特征又将病毒与政治(如间谍软件)、军事(如信息武器)、经济(如银行黑客)，以及人的行为(如购买盗版)、意识(如版权意识)、心态(如恶作剧程序)等诸多社会系统联系起来。总之，我们可以认为计算机病毒是一个复杂系统。

我们这里建立基于复杂系统理论的数学模型，主要研究的是计算机病毒的传播问题。而对病毒的破坏性、社会影响等问题，仅在对研究传播问题有反馈作用时加以分析和说明。

2.2 基本假设

2.2.1 计算机病毒的性质

现实中的计算机病毒种类繁多，行为复杂。为便于简化建模分析和计算机仿真，我们假设存在如下一种行为确定的病毒，建模时也仅研究该种病毒单独作用时的影响。

- (1) 病毒仅通过网络传播；
- (2) 一台计算机感染病毒后，只要与其它计算机进行通信，随即向对方发送病毒复本，试图使对方染毒；
- (3) 没有染毒的计算机在收到病毒后，如果不具备对病毒的防御能力，就会染毒。
- (4) 已经染毒的计算机在收到病毒后不会重复染毒，但病毒清除后如果没有采取防范措施可能再次染毒；
- (5) 病毒在计算机中如果没有被及时发现和清除，在经过一定时间的潜伏期后，会破坏该计算机的软件系统。此时认为造成了一定的经济损失。但软件系统可以快速重建，因此这台计算机并不退出病毒传播系统，而是成为一台没有染毒的计算机。
- (6) 可以通过设立硬件防火墙、安装杀毒软件等方式使计算机具有对病毒的防御能力。

2.2.2 计算机病毒传播途径

计算机病毒的传播途径主要有文件传播、磁盘引导区传播、网络传播等。文件型病毒和宏病毒主要通过文件传播，引导型病毒主要通过磁盘引导区传播，木马病毒、脚本病毒和蠕虫病毒主要通过网络传播。在上世纪九十年代中期以前，文件型病毒、引导型病毒通过磁盘复制文件传播是病毒传播的主要方式。在九十年代末，宏病毒曾叱咤一时，但随着其生存宿主——Office 系列软件安全性能的改善而基本上销声匿迹。本世纪以来，随着 Internet 的飞速发展，木马病毒、脚本病毒和蠕虫病毒成为病毒的主流。据北京信息化工作办公室及国内知名反病毒厂商瑞星公司发布的相关报告，2005 年国内上截获的计算机病毒九成以上通过网络传播，前十大影响力的病毒全部为网络传播的病毒。因此我们建立的模型仅考虑病毒通过网络传播的情况，忽略其它次要途径。同时我们也忽略联入网络的计算机数量的变化。另外由于病毒程序短小精悍，可以不考虑接入方式和网络延时的影响，认为网络上任意两台计算机在需要的时候可以立即成功地通信，如果通信一方染毒就会立即将病毒复本发给对方。

2.2.3 计算机

基于上述假设，我们首先认为所有接入网络的计算机为计算机病毒传播的主体。事实上由于计算机操作系统的差别，一种病毒一般不可能在安装不同操作系统的计算机间传播。目前全球最流行的操作系统是微软的 Windows 系列，它占有 80% 以上的桌面终端市场份额和半数的服务器市场份额。Unix/Linux 系统操

作系统是主流的服务器操作系统，考虑到目前针对 Unix/Linux 系统的病毒数量稀少，而一般的服务器软硬件安全性能有特殊配置，我们在研究病毒传播时忽略这部分主体。对于其它非主流的操作系统，它们的用户群和针对它们的病毒都更少，也忽略不计。因此我们只考虑安装 Windows 系列操作系统的计算机。

2.2.4 计算机用户

对病毒的发现、清除和防范工作都要由计算机用户来做，因此我们假定每台计算机均存在一个实时控制管理它的用户。用户的技术水平是不同的，计算机感染病毒一定时间后才会被用户发现；用户对病毒问题的认识有差异，对相关部门病毒预警和杀毒软件升级的重视程度不同；用户是理智的，在发现病毒后愿意且能够立即清除之（计算机病毒不同于生物病毒，只要掌握了确定的方法，清除很快。即使是清除方未知的新病毒，我们也认为用户会通过重新安装操作系统等方式很快解决。因此清除病毒所需时间不计）。以上几点，我们统称为用户素质。

2.2.5 其它外部因素

我们假设某种病毒在足够多的计算机上发作时才会引起有关部门（如公安部门和杀毒软件厂商）的重视，并随即发布病毒预警信息以及清除或防御病毒的解决方案。这些信息发布之后是持续存在的，计算机用户如果注意到这些信息，就会对自己的计算机进行处理，使之具有防御能力。同时我们假定在为计算机赋予防御能力时如果已经染毒，病毒也会被用户清除。

2.2.6 小结

综上，我们认定计算机病毒传播的主体为所有接入网络的、安装 Windows 系列操作系统的、在特定用户控制管理下并存在一定的软硬件安全配置的计算机。每个这样的主体不间断地运行，在病毒通过网络来袭时有一定的概率感染实病毒。感染后自身成为病毒源，可以向网络上的其它计算机主动发送病毒。

为便于建模分析，我们将计算机的因素与人的因素结合在一起，为每一台计算机病毒传播主体设定以下状态参量：

染毒状态：布尔值，表示计算机染毒与否。

防御能力：布尔值，表示计算机有没有防御病毒的能力。

用户素质：取连续值。高素质的用户能够凭借经验感觉到计算机已染毒并采取措施，以避免经济损失。高素质的用户也更关心有关部门的病毒预警，以便及时地为计算机赋予防御能力。在病毒发作并重新安装操作系统后，高素质的用户更有可能意识到问题所在，会马上赋予计算机防御能力。为便于定量分析，我们设定用户素质为在 $[0, 1]$ 区间变化的概率值，表示计算机出现上述三种状况时用户采取相应措施的概率。俗话说“吃一堑，长一智”，用户素质会因为该用户计算机病毒发作造成经济损失而有所上升。

我们再假定初始状态所有计算机均没有防御能力，并且网络上有一定比例的计算机已经染毒。假定所有用户初始状态的素质相同，为一定值。

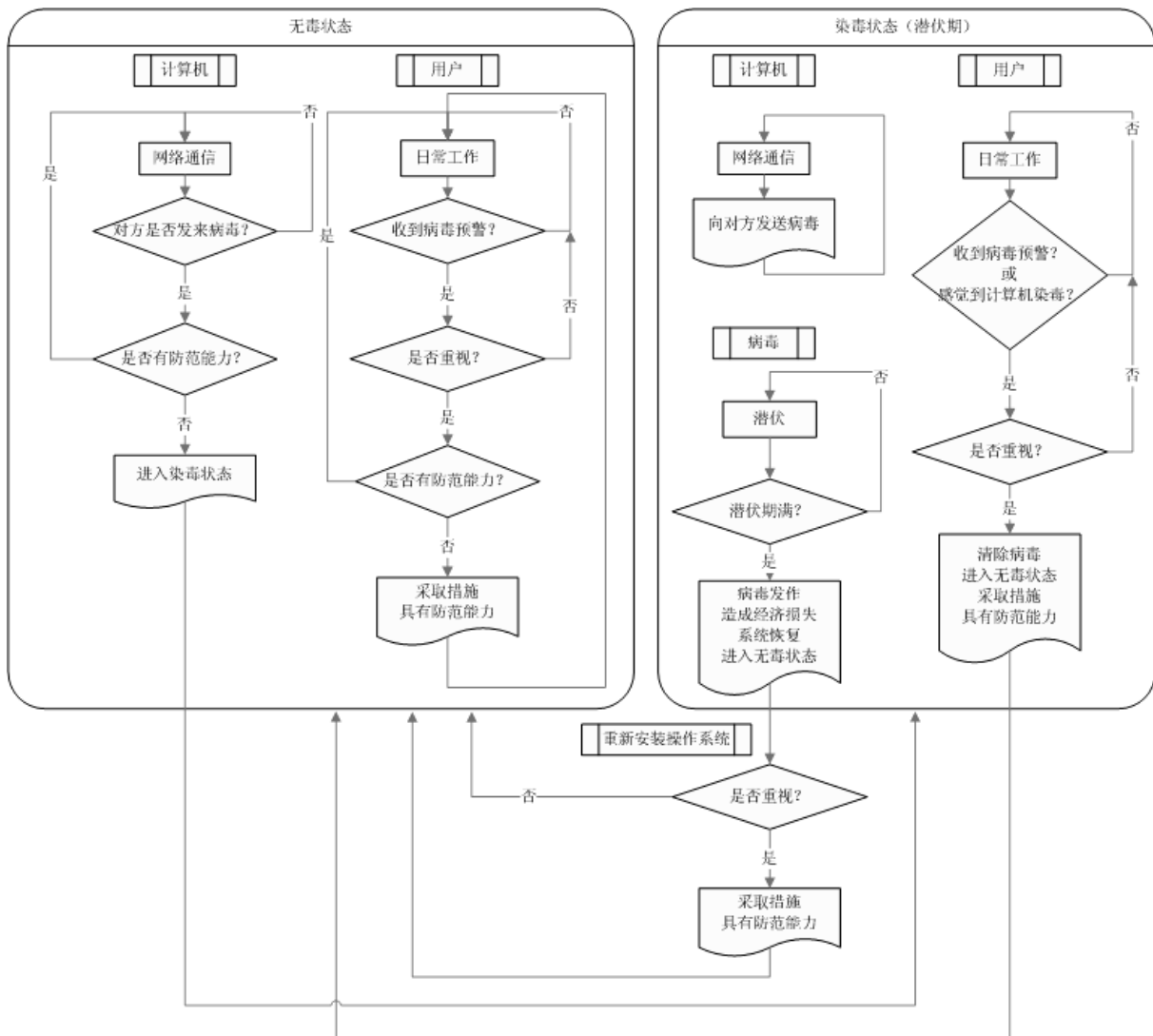
2.3 符号说明

N_a	网络上的计算机数目
-------	-----------

Nv_n	n时刻网络上染毒的计算机数目
d	病毒的潜伏期时长
u_n	n时刻特定用户素质
i	用户素质因为该用户计算机病毒发作造成经济损失时上升的值
v	初始状态网络上染毒计算机占有所有计算机比例
a	病毒预警发布时网络上染毒计算机占有所有计算机比例
c	单位时间发生通信的计算机占有所有计算机比例

2.4 模型的建立

依据上述基本假设，我们用流程图表示一台计算机由其通信过程、用户操作和病毒行为可能形成的各种状态，建立状态演化模型以便计算机编程仿真。注意其中用户“是否重视？”判断框取“是”的概率就是所谓的用户素质值。



从流程图我们可以看出，即使是经过简化的病毒传播模型，也涉及到计算机原理、病毒机理和用户的素质等多方面因素，具有多个状态变量，一定的反馈结构和由网络提供的交错繁杂的输入、输出流，可以

看成是一个复杂系统。

该系统的状态演化方程可简写为：

$$\begin{cases} Nv_0 = Na \cdot v \\ Nv_n = Nv_{n-1} \cdot q_n, n > 0 \end{cases}$$

$$q_n = f^{(n)}(d, u_n, i, a, c)$$

q_n 是由诸多状态因子决定的状态转移因子。其中 u_n 为随时间动态变化的特定的用户素质，其余参数为常量。 $f^{(n)}(\cdot)$ 是一个高阶、非线性、时变、多回路的微分函数，表示计算机病毒传播模型这一复杂系统。由于状态因子的多元性和时变性，确定 $f^{(n)}(\cdot)$ 的具体形式比较困难，因此我们采用计算机仿真方法求解该模型的数值解。

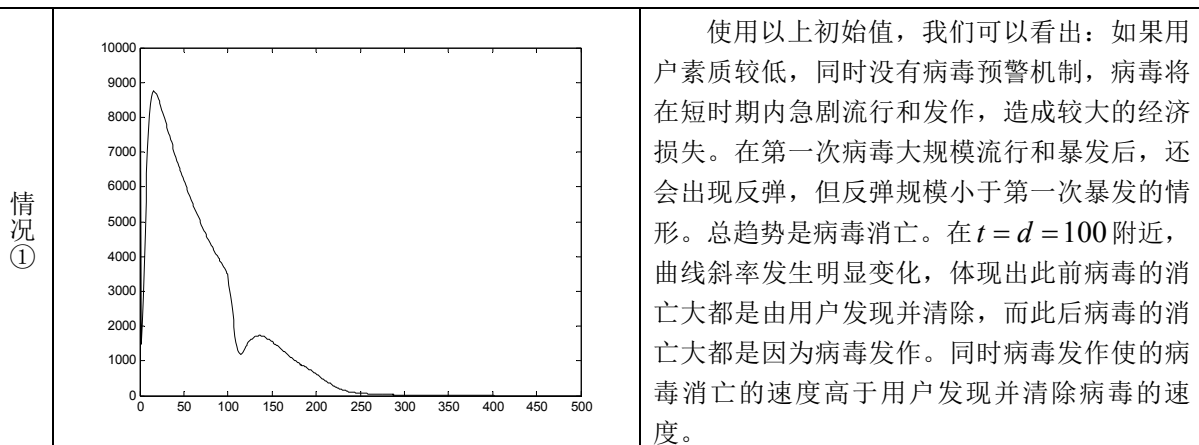
2.5 模型的求解

为直观显示计算机病毒传播过程，我们用 C++ 编程模拟上述病毒传播流程（源程序见附录），输出染毒计算机数目与时间的关系。输出结果导入 Matlab，绘制成 $Nv-t$ 曲线。

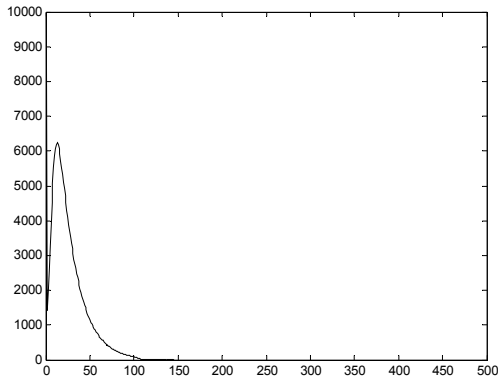
我们首先假定一些变量的初始值。

Na	10000
Nv_n	1000
d	100
u_n	0.01
i	0.01
v	0.1
a	>1（表示没有病毒预警机制）
c	0.2

在此基础上，我们经过多次试验，修改各个变量的值，得出各种情况下计算机模拟的 $Nv-t$ 曲线。下面给出一些典型曲线：

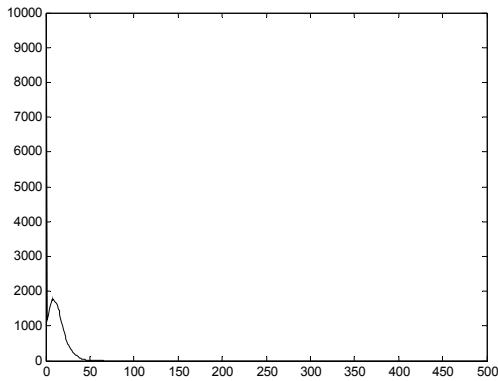


情况②



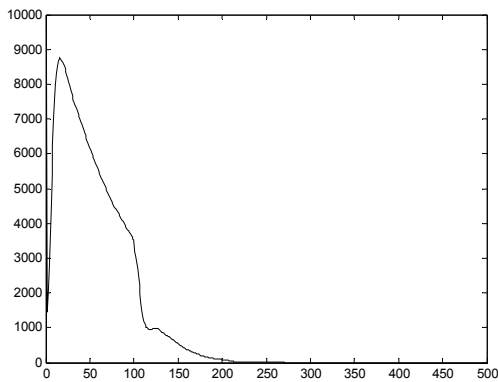
稍微提高用户素质的初始值，设 $u_0 = 0.05$ ，发现病毒同样会在短时期内急剧流行，但影响的范围会减小，时间也会缩短，且没有明显的反弹现象。多数病毒个体能够在潜伏期 d 内被发现和清除，不会造成较大的经济损失。用户发现并清除病毒的速度高于病毒发作使的病毒消亡的速度。事实上在多次仿真试验中我们发现初始用户素质变化对病毒传播趋势的灵敏度很高。

情况③



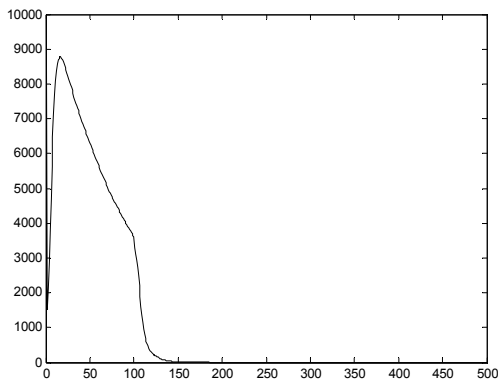
在 $u_0 = 0.2$ 时，病毒小范围内有所传播，染毒计算机个体数量不会翻番，而且病毒在潜伏期 d 内能够完全被消灭，不会造成经济损失。可见用户素质对病毒能否大规模传播有至关重要的作用。

情况④

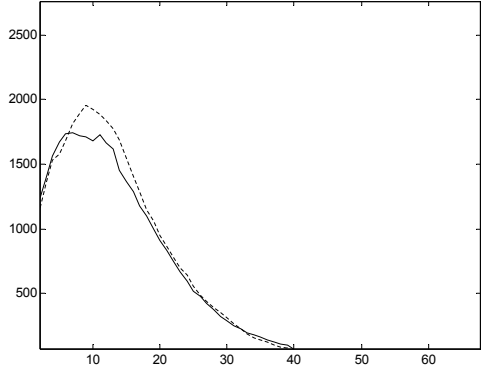
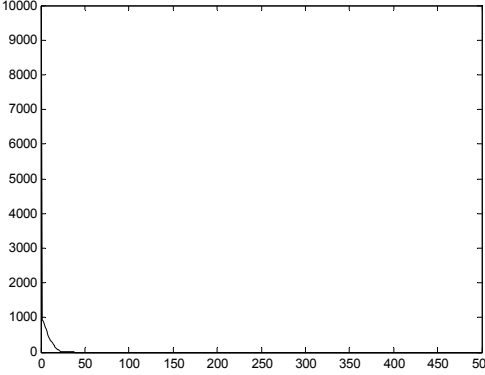


现在仍设用户素质的初始值 $u_0 = 0.01$ ，再设 $a = 0.15$ ，表示有 15% 的计算机感染病毒时有关部门就会发布病毒预警。此时病毒仍然会在短时期内急剧流行和发作，说明即使建立完善的病毒预警机制，得不到广大计算机用户的重视，也是无济于事的。但这种情况下，病毒反弹趋比较微弱，显示出病毒预警机制对抑制病毒再次传播仍有一定的作用。同时也体现出“吃一堑，长一智”因子 i 的积极作用。

情况⑤



其它条件同情况④，将 i 因子增至 0.05，可以看出病毒在暴发一次后的反弹现象被完全消除了。再次体现出用户素质对病毒传播的重要影响。

<p>情况⑥</p>		<p>其它条件同情况④，设 $u_0 = 0.05$，这时的曲线形态与情况②时基本重合，这里略去不画。这也是因为用户重视程度不够造成的，只是多数病毒个体能够在潜伏期 d 内被发现和清除，病毒预警机制和 i 因子的作用没有体现出来。</p> <p>随着用户素质的提高，他们会关注病毒预警信息，使病毒传播范围减小。情况⑥是其它条件同情况④，改设 $u_0 = 0.2$ 时的曲线（实线）与情况③曲线（虚线）的局部放大对比。</p>
<p>情况⑦</p>		<p>随着初始用户素质的继续提高，用户自身发现病毒的概率也不断增大，在达到某个阈值（本实验条件下测试出阈值 $u_0 \approx 0.35$）后，曲线的趋势变为单调递减，表示病毒还没有来得及传播，就已经被警觉的用户们扼杀了。此时若 $a > \nu$，则病毒预警机制失去作用。即使 $a \leq \nu$（现实中很少有这种情况），a 值的变化对曲线的灵敏度也较低。</p>

2.6 模型的结论

我们通过多次试验，用不同的初始值组合来生成 $Nv-t$ 曲线，特别是对以上各个典型情况曲线进行分析，得出了以下结论：

- (1) 在初始状态网络上染毒计算机占有所有计算机比例一定的情况下，初始用户素质的高低是决定计算机病毒传播速度、范围和是否造成大规模经济损失的最主要原因。用户素质对计算机病毒传播系统的灵敏度最高。初始用户素质不断提高时，病毒传播 $Nv-t$ 曲线由多次反复型变为单次起落型，再变为单调递减型。
- (2) “吃一堑，长一智”效应在初始用户素质处于一定区间内时有明显的作用，主要体现在病毒如果已经造成了大规模经济损失，广大计算机用户会产生“亡羊补牢”的意识。这种意识并不能抵制病毒的首次暴发，但能防止病毒传播的反弹现象。
- (3) 病毒预警机制在用户素质处于一定区间内时有明显的作用，也主要体现在防止病毒传播的反弹现象方面，对抵制病毒的首次暴发的影响力不大。用户素质过低，病毒预警得不到重视；用户素质升高后足以自己发现和解决病毒问题，无须预警机制来提醒。

总之，要预防计算机病毒的传播，削弱计算机病毒的危害，提高广大计算机用户的素质是关键。须知计算机病毒技术并不神秘。随着编程语言的不断高级化、简单化、专用化，目前网络上的病毒不少是诞生自普通计算机用户的键盘下的“低技术含量”病毒，有一定经验的用户都可以自行发现并清除。网络上病毒横行，很大程度上是用户安全意识不强、技术水平不高造成的。据中国互联网信息中心（CNNIC）统计，国内网民中青少年占 51.7%，他们大都没有系统地学习过计算机信息安全知识，上网工具主要是家庭及网吧中安全性能并不高的计算机，况且大都使用盗版的操作系统与杀毒软件，病毒泛滥可想而知。学习一些

计算机病毒基础理论，了解病毒破坏和传播的机理，对广大计算机用户是很有必要的。相关主管部门应加强计算机病毒知识普及，增强群众的信息安全意识，以人为本地解决计算机病毒这个社会问题。此外，病毒预警机制作为在一定范围内有效的病毒遏制方案，也应当坚持和强化。在尽早的时间内发现新的计算机病毒，将其扼杀在造成恶果之前，是最好不过的。病毒技术和反病毒技术这对矛盾在不断的斗争中持续发展，是信息社会安全领域的重要课题。动员广大人民群众的力量，严打病毒，保障信息安全刻不容缓。

2.7 模型的讨论和改进

用计算机编程仿真病毒传播的流程，充分利用了计算机强大的逻辑功能和计算能力，是一种初等的、容易理解的复杂系统分析方法。在被模拟的流程发生变化时，只须修正程序的局部，无须重新建立模型。与传统的微分方程乃至更复杂的数学模型相比，它更注重问题本身的性质，对数学理论的要求相对较低，同时对各类现实问题的适应性和应变性强，是应用领域值得推广的方法。

该模型比较成功地模拟了计算机病毒在不同条件下的传播趋势，并给出了合理的解释和可行的解决方案，但仍有一定的缺陷。例如使用用户素质这一单一因素去衡量用户发现病毒的概率、重视病毒预警的概率等多个概率值，不太符合现实。解决这个问题的方法可以是设定多个独立的概率值对应用户不同方面的素质，或者可以设定用户不同方面的素质相对于其综合素质的权值。

此外，使用神经网络模型分析本问题也是一个较佳的选择。可以将用户发现病毒与遭受损失作为奖励因子与惩罚因子作用于用户素质。这样也可以避免建立多因子的、高阶的、非线性的微分方程。神经网络模型同样可用计算机进行仿真，比直接编程模拟更具数学性，在灵敏度分析等方面有较成型的理论，无须使用大量的初始数据进行手工测试。

3 附录

3.1 计算机仿真程序源代码

```
#include <stdio>
#include <stdlib>
#include <cmath>
#include <ctime>
#include <iostream>

using namespace std;

const int MAXtime = 500;
const int NUM = 10000;
const int delitescence = 100;
const double initVirusedRate = 0.1;
const double initUserLevel = 0.01;
const double incUserLevel = 0.01;
const double alarmRate = 2;
const double communicationsRate = 0.2;

int CURdamnify = 0;
bool alarm = false;

class Computer
{
public:
    bool Virused;
    bool Immuned;
    double UserLevel;
    int Past;
    Computer() : Virused(false), Immuned(false), UserLevel(initUserLevel), Past(0) {}
    ~Computer() {}
};
```

```

};

int main(int argc, char *argv[])
{
    Computer * com = new Computer[NUM];
    int i, obj1, obj2, t, count;
    for (i = 0; i < NUM; i++)
    {
        if (i < initVirusedRate * NUM)
            com[i].Virused = true;
    }
    srand((unsigned)time(NULL));
    for (t = 0; t < MAXtime; t++)
    {
        count = 0;
        for (i = 0; i < NUM; i++)
        {
            if (com[i].Virused)
                count++;
        }
        cout << count << ',';

        for (i = 0; i < NUM; i++)
        {
            if (alarm && (rand() % 1000 < com[i].UserLevel * 1000))
            {
                com[i].Virused = false;
                com[i].Immuned = true;
                com[i].Past = 0;
            }
            if (com[i].Virused)
            {
                if (rand() % 1000 < com[i].UserLevel * 1000)
                {
                    com[i].Virused = false;
                    com[i].Immuned = true;
                    com[i].Past = 0;
                }
                else
                {
                    com[i].Past++;
                    if (com[i].Past >= delitescence)
                    {
                        com[i].Virused = false;
                        com[i].Past = 0;
                        if (rand() % 1000 < com[i].UserLevel * 1000)
                            com[i].Immuned = true;
                        com[i].UserLevel += incUserLevel;
                        if (com[i].UserLevel > 1.0)
                            com[i].UserLevel = 1.0;
                        CURdamnify++;
                        if (CURdamnify >= alarmRate * NUM)
                            alarm = true;
                    }
                }
            }
        }
        for (i = 0; i < communicationsRate * NUM; i++)
        {
            obj1 = (rand() % NUM);
            obj2 = (rand() % NUM);
            if (com[obj1].Virused && !com[obj2].Immuned)
                com[obj2].Virused = true;
            if (com[obj2].Virused && !com[obj1].Immuned)
                com[obj1].Virused = true;
        }
    }

    return EXIT_SUCCESS;
}

```

3.2 对复杂系统的介绍性短文

飞鸟如何聚集成群？经济如何出现危机？生命如何起源？战争如何爆发？这些看似不相关的问题，有一个共同的特点，就是“复杂性”。那么什么样的问题是“复杂性问题”？也许你会说：不简单的问题就是复杂性问题。然而事实可能远没有这么简单。

我们生活的世界可以认为是由各种大大小小的系统构成的。它们相互影响、相互依赖，形成一定的结构与层次。系统从其内子系统的关联关系角度可划分为简单系统与复杂系统。简单系统大家都不陌生，我们在基础物理等学科中接触的大都是简单系统，例如封闭的气体、遥远的星系和行驶的汽车。它们通常只具有少量的研究对象，对象之间的相互作用比较确定，对象的行为也很相似，以至于我们能够应用简单的公式或统计方法来精确地研究它们的行为。而在现实生活中，我们接触更多的是复杂系统，比如政治经济系统、人体生理系统、自然生态系统等。复杂系统是相对简单系统而言的，其最主要特征是系统具有众多的状态变量，反馈结构复杂，输入与输出呈现非线性特征。这些复杂系统和复杂性问题系统在结构、功能、行为、演化等方面十分复杂，至今仍有大量问题还不被了解。即使建立了求解模型，也往往得不到精确的结果。

复杂系统涉及多个学科的，因此要从不同的学科、不同的侧面，用不同的方法和工具，进行多方面的交叉研究。由此蕴生的复杂性科学是一门具有重大理论意义及实际价值，并亟待开展研究的崭新科学。二十一世纪是知识爆炸的时代，紧随着知识爆炸到来的必然是各领域科学的交叉与融合。复杂系统理论作为知识整合的工具，是未来科学发展的必然选择。深入研究复杂系统理论，无论对科学本身的发展还是现实问题的解决，都势在必行。

参考文献

- [1]李绍良, 复杂系统的研究方法, 哈尔滨理工大学硕士学位论文, 2004;
- [2]曾禹村 等, 信号与系统, 北京理工大学出版社, 2002;
- [3]彭国军, 计算机病毒技术与反病毒, 武汉大学硕士学位论文, 2003;
- [4]韩筱卿 等, 计算机病毒分析与防范大全, 电子工业出版社, 2006