CHEERS
湛庐

# 为什么需要生物学思维

SAMUEL ARBESMAN
[美] 塞缪尔·阿贝斯曼 著

# OVER-COMPLICATED

## 洞悉复杂世界的思考方式

## TECHNOLOGY AT THE LIMITS OF COMPREHENSION

贾拥民 译

圣塔菲书系 Santa Fe Institute Series

四川人民出版社

# 版权信息

本书纸版由四川人民出版社于2019年5月出版

作者授权湛庐文化（Cheers Publishing）作中国大陆（地区）电子版发行（限简体中文）



书名：为什么需要生物学思维

著者：塞缪尔·阿贝斯曼

电子书定价：62.99元

Overcomplicated: Technology at the Limits of Comprehension by Samuel

Arbesman

此书送给

将在一个奇异世界中长大的

阿比高尔和内森

请永远不要失去对这个世界的激情

# 引言　认识复杂系统

2015年7月8日，我还在撰写本书的时候，美国联合航空公司的计算机系统出了故障，所有飞机被迫停飞。同一天，纽约证券交易所的计算机系统也遇到了问题，所有交易被迫中止。《华尔街日报》的网站也崩溃了。接二连三的系统问题令无数人惊慌失措，但是没有人知道到底发生了什么。Twitter上谣言四起，许多人都在猜测：美国是不是遭到了网络攻击?

但是，这些似乎并不是有预谋的网络攻击。“罪魁祸首”更有可能是问题很多且尚未被人完全掌握的软件系统。[1]一位安全专家在评论该日事件时所说的话可谓一针见血：“这些都是极其复杂的系统，出错的方式多种多样，其中有很多是我们目前完全无法理解的。”[2]事实上，对于这些复杂得令人难以置信的技术，说我们根本无法尽知其出错方式，其实已经是在有意地轻描淡写了。

人类的技术，从网站到交易系统，从城市基础设施到科学模型，甚至是为大型企业提供配套服务的供应链系统和物流系统，都已变得过于复杂且相互交错。在很多情况下，就连当初构建、每天使用和维护它们的人也无法完全理解了。

在《创造力差距》（The Ingenuity Gap）一书中，加拿大政治学家、生态学家托马斯·霍默－狄克逊（Thomas Homer-Dixon）描述了他于1977年在法国斯特拉斯堡（Strasbourg）参观高能粒子加速器时的情景。[3]当时，他问一位在那里工作的科学家，想知道是否有人明白整台加速器到底有多么复杂，结果得到的答案是：“没有任何人能够完全搞懂这台机器。”后来，霍默－狄克逊回忆说，这个答案令他深感不安。是的，我们理应感到不安。从那时起，粒子加速器以及人类所构建的其他一切事物的复杂程度，几乎都在不断增加。

技术的复杂性一直在增加。以铁路为例，铁路出现后，轨道网络和调度系统应运而生，以保证列车能够安全且准点运行。因为要对遍布整个大陆的无数列车进行调度，所以美国的标准化时区制也得到了发展。在铁路技术诞生之前，或者说在其复杂情况接踵而至之前，标准化时区制本不是必需品。

然而时至今日，技术复杂性已日益接近一个临界点。计算机的出现又给环境平添了许多激进新颖性（radical novelty）[4]，这是计算机科学家艾兹格·迪科斯彻（Edsger Dijkstra）提出的一个术语。计算机的硬件和软件比之前的任何事物都要复杂得多：一个程序可能包含数百万行计算机代码，微芯片内集成微型元件的尺寸也已经小到接近原子量级。现在，计算机已经被应用到了汽车、手机，以及金融市场等领域，而不断增加的技术复杂性也已经到了令人费解的程度。

近年来，科学家们甚至还逐渐认识到，技术已与自然密不可分。研究地球岩层的地质学家已经在正式考虑这样一个问题：是否有足够的证据支撑我们把当下这个时代命名为“人类世”（Anthropocene）[5]，也就是人类的时代。无论正式名称是什么，我们所制造的人工系统与自然世界之间的亲密关系都意味着，人类的每一个行动都比以往任何时候更有可能引发意想不到的后果，其影响不仅将涉及生活的方方面面，还将波及地球的每个角落，甚至在某些时候会超出地球本身。

从整体上看，技术和基础设施已经变成了极其复杂的、物理化和数字化的系统，好似血液循环系统那般，牵引着地球上所有原材料的流动，同时“排放”出道路、摩天大楼、芸芸众生，以及化学废水。技术加速了地球的“新陈代谢”，而这个过程是在一场极其复杂的“材料之舞”中完成的，这一切甚至改变了地球表面的光芒。

面对这种复杂性，我们往往会情不自禁地发出心难两用的感慨。一方面，我们构建出了这些令人难以置信的复杂系统，这无疑是值得自豪的，尽管它们也许永远无法如预期那般正常运行，但是它们作为无比复

杂的大型系统，本身就是值得赞叹的。另一方面，我们在技术领域所付出的所有努力，几乎都在使我们远离优雅和简单，并且让我们无法抗拒地一步步走向某个极度复杂、完全无法预料的世界。

这段“旅途”的终点，其实已隐约可见，那是一个几乎可以自成一体的技术生态系统，一个超越了人类认知和理解能力的极限世界。正如2013年9月《科学报告》（Scientific Reports）所刊发的一篇论文指出的那样，现在，一个完整的“人类不可能及时应对的新机器生态”已经出现了。[6]需要注意的是，这篇论文所讨论的范畴还仅仅是金融领域而已。在股票市场上，各种机器通过各种形式进行着交互，从根本上说，这与利用不同算法进行交易没什么两样，而人类则成了局外人。[7]

本书的观点是，如今存在着这样一种趋势和力量：无论人们做什么，都会使技术日趋复杂且变得难以理解。这种趋势和力量意味着，2015年7月8日那天所发生的事件将会越来越多。人们原本认为非常可靠的系统会发生无法解释的故障，甚至崩溃！

作为一名研究复杂性的科学家，我花了很多时间研究生活中日益增加的复杂性。我早就注意到，在巨大的复杂性面前，普通人一般会做出两种反应，这两种反应都因各自的极端化而有失偏颇：第一种反应是对技术所导致的复杂世界满怀恐惧；第二种反应是对技术过度崇敬且深信不疑。

恐惧是一种非常自然的反应。长久以来，人类一直被“淹没”在各种各样的末日预言中：大型机器会变成杀手，超级智能计算机即将出现，程序无法保证自动驾驶汽车不会撞死行人，等等。这些技术都非常复杂，即使是专家也无法完全理解，碰巧它们又相当可怕。这样一来，我们在接近技术、使用技术的时候便会警觉不安、惊恐不宁。

就算对技术系统本身并不畏惧，许多人还是会对已经形成包围之势的算法和技术如履薄冰，甚至感到厌恶，不愿直面技术所拥有的惊人力量。当亚马逊或奈飞推送购买建议时，我们无从拒绝；当敲下的文字被

输入法自动纠错时，我们恼怒不已，这些都是我们的切身体会。甚至，在面对应用程序所推荐的从一个地方到另一个地方的最佳路线时，许多人偏要反其道而行之。毫无疑问，这种“算法厌恶”现象反映出了许多人共有的一种情绪，而这种情绪其实就是“技术恐惧”[8]的表现之一，只是不那么严重而已。

与此同时，还有些人走向了另一个极端：对技术过度崇敬。在面对某个行为奇异的复杂事物时，很多人最终都会选择到宗教中寻求慰藉。当领教了谷歌公司“大脑”的“博学”[9]，也就是对用户需求的精准预测之后，抚摸着心爱的最新款苹果手机时或访问一个庞大无比的数据中心时，我们的内心可能会蠢蠢欲动，感觉自己像是走进了一座庄严雄伟的大教堂。这时，崇敬之心油然而生。

然而，这两种反应，无论是来自专家的，还是来自非专业人士的，皆不恰当，或者说，皆是非生产性的。在并不值得我们深信不疑的系统面前，无论是恐惧也好，崇拜和敬畏也罢，都会阻碍我们选择恰当的面对方式。若无法采取切实到位的措施，我们就有可能失去控制权，而这会带来意想不到的、甚至是毁灭性的后果。当系统再度崩溃时，这种“理解不足”所导致的问题可能就不会再像《华尔街日报》事件中那样，仅仅只是读者无法在网站上阅读文章那么微不足道了。故障可能会发生在电网、银行系统，甚至医疗领域中，而且不会自行消失。我们现在如果不正视它们，未来定会陷入险境。

技术无处不在，既不古老也不深奥，说到底，它源自某些完美的无限遐想。[10]纵然技术是十分混乱且不完美的，但它却很“平易近人”。我们所需要的是应对各种情况的策略。

我写这本书是为了帮助人们在上述两个极端反应，也就是恐惧和崇敬之间找到一条正路，并为接近技术、理解技术和掌控技术探索出方向。那是一个乐观的方向，能够改变对这些系统的思考方式，让我们不会陷入恐惧或崇敬，不会陷入“六神无主”。

想要朝这个方向出发，就得向技术“妥协”，而这又要求我们得先培养出一种面对技术系统时的舒适感，尽管我们从未完全理解它们。这是一种谦卑的舒适感，既要容忍不确定和不完美，又要不断努力探索，以期逐渐了解更多。正如即将看到的那样，除了其他方面，我们在探索这个方向时，还需要借鉴科学家研究生物复杂性时所采用的方法。

尽管人类过于依赖技术系统，而这些系统又过于复杂，但我仍然相信，人类终有希望掌控自己所构建的系统。

本书旨在说明为什么系统会过于复杂，以及我们应该怎样理解和掌控过于复杂的系统。

扫码下载“湛庐阅读”App，

搜索“阿贝斯曼”，

获取作者演讲视频及更多精彩内容。

# 目录


连接和反馈是复杂系统的典型标志。技术系统已经变得如此复杂，以致每个专家都仅知道其中的一个部分，没有人能够完全理解整体。我们已经从“启蒙时代”迈进了“纠缠时代”。

1986年初的一个冬日，距离“挑战者号”航天飞机爆炸事件发生还不到一个月，著名物理学家理查德·费曼（Richard Feynman）在调查委员会的听证会上公开了一份调查报告。[1]费曼非常冷静地揭示了导致“挑战者号”航天飞机起飞后不久就爆炸解体的原因。问题出在用于密封固体火箭助推器各部分之间缝隙的O型圈上。O型圈其实是一小片橡胶，放入冰水后会失去弹性，也就是说，这种小部件对温度变化非常敏感，所以无法保证密封的牢固性。这些O型圈很可能就是这起导致7名机组人员丧生灾难的罪魁祸首。

与此事件如出一辙的是发生在汽车业中的另一事件。2007年的一

天，琼·布考特（Jean Bookout）驾驶着一辆生产于2005年的丰田凯美瑞，行驶在俄克拉何马州的一个小镇上。突然，她的车开始加速，并失去了控制。[2]她踩下了刹车，但没有任何作用；她拉起了紧急制动器，轮胎在道路上留下了触目惊心的摩擦痕迹。然而，这些努力都无济于事，汽车最终狠狠地撞到了路堤上。布考特受了重伤，车上的乘客、她的朋友芭芭拉·施瓦茨（Barbara Schwarz）则不幸遇难。

并不是只有布考特一个人遇到这种怪事。事实上，多年以来，在丰田公司生产的汽车中，有多种型号的汽车都曾出现过此类奇怪且危险的故障：汽车在行驶过程中会“违背”驾驶员的意愿保持匀速，甚至还会“擅自”加速。这种意外加速事故已经导致多人伤亡。业内专家分析出了这类事故的几个导火索：驾驶员操作失误，垫毡顶住了油门加速踏板，油门加速踏板黏滞未能及时复原，等等。但是，这些原因并不能解释所有的意外加速事故。在受事故影响的车型中，因为垫毡或油门加速踏板不合格而被召回的并不多，而且我们也没有理由认为，丰田汽车的驾驶员比其他汽车的驾驶员更容易犯错。

无奈之下，丰田公司请来了嵌入式软件专家迈克尔·巴尔（Michael Barr）追查原因，并向他开放了专有数据库以及向来严格保密的软件代码。在6位经验丰富的工程师的协助下，巴尔几经周折终于找到了问题的根源。与此同时，计算机科学家菲利普·库普曼（Philip Koopman）也通过对公开程序的仔细分析，找到了事故发生的大致原因。[3]两位专家都认为，对于意外加速事故的发生，丰田发动机软件系统的过度庞大和极度复杂，以及糟糕的设计都应该承担相应的责任。我们无法将事故责任明确地归咎于某个设计或部件出了错，毕竟，这里面存在的问题盘根错节，而且这些问题还会引发汽车软件系统与外部机电系统之间的巴洛克式结构（baroque structure）的大规模交互。无论是单独看来，还是整体看来，这个系统的极度复杂性让我们很难理解这些相互作用、相互影响的部件的深层问题和缺陷。有依据表明，丰田公司本来完全没有必要搞出一个如此复杂的系统，至少在现有

情况下，不必如此复杂，它们显然不够小心。[4]

这个“诊断结果”与我们熟悉的“技术失败”并无二致。费曼找到了导致“挑战者号”航天飞机爆炸的单一原因，有所不同的是，我们无法确定导致丰田汽车出现事故的单一原因。无论如何，如果部件堆积得越来越多，设计也越来越复杂，那么灾难便只会越来越多。

事实上，即使找到了导致失败的单一原因，但在极度复杂的系统中，那很可能也只是掩人耳目的东西罢了。1996年，阿丽亚娜5型火箭在发射升空39秒后爆炸，火箭上搭载的4颗卫星均毁于一旦。[5]事故分析表明，发生爆炸的原因是，火箭在新环境下使用了一些较为陈旧的软件代码。但是，根据霍默－狄克逊教授的说法，没有任何一个承包商被追责。这次爆炸并不是某个决策失误所致，而是整个发射系统的极度复杂性所致。其他类似的灾难还有三哩岛核电站事故，[6]尽管有关方面也找到了一个原因，但是究其根本，仍然得归咎于系统的庞大和极度复杂，而非某个部件或决策出了错。[7]

当我们试图将这种复杂性拆解开来时，难免会陷入“挑战者号”航天飞机的主流事故中。[8]尽管航天飞机的发射和操作是一项异常复杂的工作，但是很多人都相信，只要彻底核查复杂系统，将其分解开来，确定每个部件如何工作，在什么情况下会出故障，我们就应该能够理解它们。这无疑是一种过于自信的想法，其根源在于：人类的思想拥有无限潜力。人们总认为，只要足够努力，就可以完美地理解身边的一切事物，特别是人类自己所构建的那些东西。

我们观察到，这种情绪在所谓的辉格党式（Whiggish）的进步观念中体现得最为充分。正如科学作家菲利普·鲍尔（Philip Ball）所描述的那样，这种观念认为人类已经踏上了“从无知和迷信的黑暗时代走向理性时代的胜利之旅”[9]。在科学界和技术爱好者的圈子中，这个观念根深蒂固，正如历史学家伊恩·毕考克（Ian Beacock）在文章中所写：“科技行业讲述了人类‘数字时代’的崛起，那是一个辉格党式的

故事：阴郁昏暗的时代行将结束，人类将迎来一个更加美好的未来，一个更幸福、更开放的世界。在那个美丽新世界中，一切都是测量好的、经过精心设计的，并达到了最高效的完美状态。”[10]当然，坚持这种观点的人还认为，效率和生产力的提高，促使人们为了实现某个令人振奋的目标，而持续关注和理解自身所构建的现代工程。这种辉格党式的观点与社会学家马克斯·韦伯（Max Weber）所描述的“现代思维模式”有关：在一个“不再拥有幻想的世界”中，人们会产生一种感觉，也就是“在原则上，我们可以通过运算来掌握所有事物”。[11]

然而，这种理解复杂性的方法如今已经失效了。是的，我们正处在一个新时代，正在建立一个新系统，而这个系统是无法从整体上被掌握的[12]，或者说，无论对谁而言，它都太复杂了。我们发现，在分析丰田汽车意外加速事故时，无论是不是专家，上述那种陈旧的思维模式都不足以应对其复杂性了。不仅如此，这些情况不只附着在我们缺失的经验中，它们已充斥了我们的整个生活。

## 什么是复杂系统

在这里，我们必须先花些时间讨论一下“庞杂”（complicated）和“复杂”（complex）系统的含义。[13]尽管我在文中，以及书名中使用这两个术语时所指含义与它们在口语中的意义大致相同，但实际上，它们的准确含义和日常意义存在着一些重要区别。

请想象一下，有许多水上浮标被绑在一起，漂浮在水面。[14]当一艘船经过时，其尾流会形成一个个小波浪，从而使浮标一个个动起来。但是，每个浮标都无法单独“行动”。因为每一个浮标都通过绳索连接着不同重量和大小的其他浮标，所以任何一只浮标的行动，都会带动其他浮标产生相应的行动。这些行动还可能会引发意想不到的反馈过程，也就是说，浮标的行动会间接地影响到自身。于是，船的简单尾流，在这个复杂的浮标网络中引发了大量级联式（cascade）的行动。如果船以

其他方式驶过，比如以另一种速度或角度驶过，那么浮标的行动可能会完全不同。

接下来再想象一下，人们把这些浮标从水中捞了出来，扔在了码头上。它们可能被排列得非常巧妙，却很难描述清楚，我们需要颇多文字才能讲明白它们之间的相对位置，或许还需要附上一些图表，以便日后能复制出同样的排列方法。但是，无论被排列成什么样，这里也不会有趣事发生。因为在这个精巧的网络中不存在级联效应，也不存在反馈过程。说到底，只不过是一堆可以漂浮的东西被放在了码头上。

水中的浮标形成了一个复杂的系统。那么码头上的浮标呢？它们的排列也许看上去很巧妙，但其所构成的系统最多只能说是庞杂的。任何一个系统，要想成为一个复杂的系统，仅仅包含很多组成部分是不够的，各部分之间还必须相互关联，并在“嘈杂的舞蹈”中相互作用才行。在这种情况下，我们可以观察到某些行为特征，那是复杂系统的标志：微小的变化通过网络被级联式放大，并引发反馈过程，同时敏感地依赖于系统的初始状态。这些特征，与其他一些特征一起，使系统从“庞杂的”变成了“复杂的”。

对于这种区别，还可以从另一个角度来思考：生物在活着的时候是复杂的，而死后最多只能说是庞杂的。死去的动物虽不失其复杂特性，但其内部却没有任何过程在发生：整个生命网络，比如血液循环系统、新陈代谢系统、神经系统等，都很“安静”。生命是运动和相互作用的“狂暴之流”，极其复杂精妙，在有生命的机体中，再小的变化都会被级联式放大，进而引发大量行为。另外，即使一个系统是动态的，比如浮在水中的一堆没有被连接起来的浮标，因为其内部不存在相互联系，没有反馈的可能性以及其他类似的属性，所以我们也只能称其是“庞杂的”，而非“复杂的”。

如果将技术定义为人类为特定目的而设计和构建的各种类型的系统，那么就不难注意到，当今最先进的技术几乎都是复杂的系统：动

态、功能复杂、规模庞大，而且拥有近乎有机生命体般的复杂性。这些复杂的系统遍布我们周围，从汽车软件到电脑设备，再到城市基础设施。[15]那些庞大的、高度互联的软件，其规模足以与百科全书相比。据估计，微软的办公软件就有数千万行计算机程序代码。[16]

美国的公路系统有30万个交叉路口，每一个路口都有交通信号灯。[17]这是一个覆盖全美的动态交通网络，所包含的自动纠错系统先进得令人难以置信，尽管我们会因为它经常出问题而嘲笑它愚蠢得令人绝望。该系统的背后是PB级数据（1PB等于100万个GB）和复杂的概率模型。[18]随着时间的推移，法律体系也变得越来越复杂。截至2014年，美国联邦税法的总页数已经超过了74 000页。[19]这个庞大的法律网络非常复杂，拥有无数个关联点，会对纳税人产生级联式的影响，可以说几乎没有人能从整体上完全理解其功能。

在我们的生活中，复杂的技术系统无处不在，这通常是一件好事。正是在这种庞大的复杂性中，我们发现了惊人的弹性，也就是复原力。这些系统通常拥有许多功能各异的工具箱和故障保险箱，可以帮助人们完成任何“想要”完成的事情。这些系统也为我们提供了连古代皇室都无法想象的生活：免除了辛苦的重复劳动；把水电送到家中；让我们一年到头都可以生活在舒适的温度下；还能帮助我们快速地传递信息。

那么，我们该如何理解复杂系统，而这又意味着什么呢？无论是分析一个现象，还是了解一个系统，都不是非此即彼的二元选择。这种理解其实是一条内涵丰富的光谱。[20]换句话说，你可以只从整体上去理解一个系统，掌握它的全局情况，而不一定非得理解其各个组成部分的细节情况；你也可以只理解它的所有组成部分，而不太关注其整体功能；你还可以只了解各部分是如何相互关联的，或者只探究这种关联的最终效应。此外，所有的“理解”都会涉及一些具体活动：描述事物是如何运行的，在不同层面上预测其未来的行为，并在时间充足和资源丰富的情况下通过构建模型来复制它。

回到上文中有关浮标的例子。你或许只能了解到两三个被连接在一起的浮标的行为细节，甚至只能搞清楚一个浮标的行为，而无法描述所有浮标所组成的巨大网络的整体行为；或者，你能够描述浮标的整体运动，却无法预测它们各自的行为。对于软件，你或许可以很好地理解给定程序中的若干模块，例如计算圆周率值的子程序，或是对一组数字进行排序的子程序，但你却不一定能理解它们是如何一起发挥作用的。通常，我们只能把握上述“理解”中的一部分，而非全部。

另外，“理解”也不是固定不变的，它可以通过训练来改进。从来没有下过国际象棋的人，在看到棋局的时候，可能无法将势均力敌的平局和王已经受到致命威胁的残局区别开来。而接受过一定训练的初学者和中段棋手，均能掌握下棋的基本模式，并判断棋局的基本形势和大致走向。国际象棋大师则能一眼看透当前的形势和接下来可能会出现的演变，而后评估棋局并给出可靠的走法，同时弥补潜在的弱点。经过充分训练后，棋手在看到一盘棋时，眼中所见的就不再是一排排、一列列的棋子了，而是诸如“白棋三步就可将死对方”之类的棋局。因此，学习专业知识并接受一定的训练确实可以从根本上改变我们对世界的看法和理解方式。

对于人类构建的系统而言，亦是如此。计算机代码既可能是难懂的天书，也可能是解决难题的优雅方案，这一切都取决于你到底了解些什么。不过，当我们无法完全理解它们时，就会遭到某种特定的失败或打击，也就是某种意想不到的结果。

以空中交通预警防撞系统（Traffic Alert and Collision Avoidance System, TCAS）为例，人们创建这个系统的目的是防止飞机在空中相撞。这个系统会提醒飞机驾驶员注意潜在危险，并告之如何根据规则做出应对。但是，几十年来，这个系统的规则已经变得极其复杂，复杂到全世界只有屈指可数的几个人能够真正了解它。每当有人提出一项新的系统规则时，有关方面就会通过模拟实验来测试其效果。在

若干次测试中，如果表现均能达到预期，新规则就会被批准投入使用。

尽管避免飞机相撞本就是一个复杂的挑战，但是从根本上来说，我们为此而建立起来的系统已经变得太过复杂了。不但一般人无法理解，即使是专家，有时候也会对系统在某些情况下所做出的反应感到疑惑。

当结果出人意料时，就意味着我们对相关问题的理解还不够充分，因而无法搞清楚这种结果从何而来。倘若只是视频游戏中的一个错误，那倒也无伤大雅，甚至还能起到娱乐大众的效果。然而，如果是极度复杂的社会保障系统发生了出人意料的状况，我们肯定就无心娱乐了。这些复杂的系统可能是为我们提供水电的基础设施，也可能是执行金融交易的软件，还可能是防止飞机相撞的程序。在这种时候，理解是否到位就成了一个生死攸关的问题。

每个人理解世界的能力天生就有所不同。天才拥有常人难以企及的直觉，他们的跳跃性思维可能远远超出了常人所能理解的范畴。但是，人类的认知能力终归是有限的。随着时间的流逝，我们所构建的各种技术系统变得越来越复杂，系统之间的关联性也变得越来越强，越来越难以理解。无论人类有多聪明，记忆力有多强，都无济于事，因为这些系统的构建方式与人类的思维方式不一样。人类不具备同时应对数百万个组成部分及其之间的大量交互，并将所有结果都记在脑袋中的能力。我们的大脑会“严重超载”，继而宣告失败。

## 从“启蒙时代”到“纠缠时代”

在当今这个机器时代，非技术专家在试图理解技术时，偶尔会诉诸迷信或某种一厢情愿的观点。通常来说，在一个家庭中，总会有一个人会因计算机或某种机器无法正常运行而受到指责，在其他人看来，他只是用手碰了一下就把事情搞砸了。有时候，甚至只要他在场，人们就会认定他是导致某种技术无法正常发挥作用的原因。例如，一个大学生放

暑假回到家，偏巧打印机坏了；或者父母来访，电脑鼠标就失灵了。

反之亦然。一个你无法解决的问题，别人却能轻松解决。你将某个出了故障的设备交给技术人员维修，可是一到他们手上，故障就消失得无影无踪，而当你把设备带回家后，却发现它仍然无法使用。

在缺乏专业技术知识的外行人看来，机器的内部工作原理相当神秘。如果我们只是这些系统的使用者，那么不知其详也并无大碍。当机器出现故障时，我们可能会半真半假地认为有人正在对机器“发功”，对它施加了不利影响。当然，不管有没有做出这样的假设，至少我们会很自然地相信，肯定有某个专家知道到底是怎么回事，比如“主板烧了”，于是我们便将那个神秘的捣蛋鬼因素抛之脑后。

然而，持有这种“迷信”观念的，早已不再仅限于普通人了，事实上，就连技术开发者也开始这样想了。美国计算机工程师李·费尔森施泰因（Lee Felsenstein）讲述了一个发生在某位工程经理身上的故事：在对软件进行展示的时候，那位工程经理不得不选择回避，因为只要他在场，软件就会运行失常。软件工程师们完全搞不清楚为什么会这样，总之只要这位经理在场，事情就会变糟。[21]费尔森施泰因指出，这种无法通过运算来解释的失败“已经落入形而上学的领域”。工程师们不明就里，下意识地把思考点放在了“存在问题”这个层面上，并走上了哲学思考之路。

当然，他们并不孤单。计算机科学家杰勒德·奥尔茨曼（Gerard Holzmann）也有同样的感受：

> 大型的、复杂的代码总会包含一些不祥的“暗码”（dark code）(1)。没有人能完全理解这种代码，它的存在也没有明确的目的。然而，想要让应用程序按照预期正常地运行，这种代码又是不可或缺的。你不想去碰它，所以会倾向于绕过它。
>
> 暗码是会逆转的。应用程序具有防止实际代码被追溯的功能，

它会莫名其妙地做出一些程式之外的事。[22]

在司法领域中也存在类似的状况。根据律师兼作家菲利普·K.霍华德（Philip K. Howard）的说法，我们目前正处在这样的境况中：“现代法律的丛林已经太密集了，成了不可知之境。”[23]在技术领域，这样的例子显然更多，甚至不需要到最前沿去找。正如作家奎因·诺顿（Quinn Norton）所指出的那样，即使是功能平庸的家用台式电脑，也是“非常复杂的，复杂到地球上的任何人都不知道它在做什么、怎么做”。[24]

近几十年来，在人类一手打造的“创世界”中，异常庞杂、难以理解的事物越来越多。事实上，即使是这些系统的创造者也常常一头雾水。美国政治理论家和技术哲学家兰登·温纳（Langdon Winner）在其著作《自主的技术》（Autonomous Technology）中指出，英国著名小说家和政治家赫伯特·乔治·威尔斯（H. G. Wells）晚年提出的那个观点是值得相信的：“人类的思维已经无法应对它自己所创造的环境了。”[25]威尔斯早在1945年就得出了这个结论，不过，他当时讨论的主题是“人类的组织和社会本身”。近年来，这个问题变得愈加尖锐，尽管技术已经发展到当年的威尔斯完全想象不到的水平。

计算机科学家丹尼尔·希利斯（Daniel Hillis）认为，我们的世界已经从“启蒙”（enlightenment）转向了“纠缠”（entanglement），至少技术领域肯定如此：“技术已经变得如此复杂，以致我们无法完全理解它，也无法完全控制它。我们已经进入了‘纠缠时代’……每个专家都只了解难题的片段，却无法把握难题的整体。”[26]就连作为技术创造者的专家都无法完全了解技术了。

## 抽象的局限

在创建复杂的技术时，最强大的方法就是人们常说的“抽

象”（abstraction）。从根本上说，抽象其实就是：将系统中某些不必要的组件细节隐藏起来，同时保留组件与系统进行高效交互的方式和能力。例如，我在编写计算机程序时，不必再用机器代码语言来编程，而可以使用C语言或其他类似的语言。机器代码语言是一种二进制代码，此前，每台特定的计算机都需要用这种语言来设置指令。现在，我写出来的程序不仅易于阅读，还可以转换为机器代码语言。在很多情况下，我甚至不需要知道程序将会在哪种特定的机器上运行，那些与机器有着更深层级交互的其他程序自然会“考虑”细节。换句话说，这些细节在编程时已经被我抽象化了。

这种抽象方法在技术中无处不在。在与某个界面友好的网站进行互动时，对于网站的内部技术细节，我们并不关心；在将烤面包机的插头插入某个插孔中时，我们也不需要知道电力是哪里输送过来的，或是发电厂位于什么地方。这就好比我们无须知道搜索引擎是如何给出具体结果的。只要接口或界面是合乎逻辑的、可以使用的，那么我们就只需要关注正在构建或修复的细节，而不用担心接口和界面背后的一切复杂问题。利用这种抽象方法，我们可以在一种技术的基础上构建出另一种技术，也就是直接利用他人创建的技术，而无须了解其内部细节。如果你是使用统计软件包分析数据集的财务分析师，或是使用预设代码创作有趣动画的应用程序开发者，那么你无疑已经在使用抽象方法了。

抽象给人们带来的最大好处是专业化。即使系统拥有数百万量级的交互，系统的构建者和维护者也不是必须知道它到底是如何工作的。抽象使他们只需了解自身关注的某个具体部分，而其他部分的细节会被再次抽象化。

然而在这个纠缠时代，抽象方法也可能会崩溃。事实上，这种情况出现得越来越频繁。在一个系统中，原本被设计者和构建者屏蔽的各个组成部分，正在越来越多地以意想不到的方式发生碰撞。

这一点在金融领域显得尤为突出。在当今的金融市场上，“参与

者”早就不仅限于人类了，大量以各种信息为基础的计算机程序也参与了交易，而且速度比人类手动执行快了无数倍。这些计算机程序以异常复杂的方式相互联系着，并通过巨大的交易网络对决策进行级联式放大和传播。那么，它们究竟是如何做出交易决策的呢？某些计算机程序能从海量的数据中总结出有意义的参数。

结果可能会非常极端。2010年5月6日，全球金融市场出现了闪电崩盘，股市出现了大规模的、非常迅速的巨幅震荡。[27]许多上市公司的市值都因此遭受了重创，不过不久之后又都基本重回原位。这次闪电崩盘涉及一系列交易算法和实施细则，这些算法和细则以意想不到的方式进行了交互，在短时间内便造成数十亿美元的损失。尽管很复杂，但这些系统并非处于真空地带，它们是更高层级的技术生态系统的一部分，而这个技术生态系统决定了每种证券或商品的交易时段。金融系统还会受到一系列法律法规的约束。当然，法律法规本身即是一个系统，而且是一个庞大且复杂的系统。不同法律之间存在相互依赖、相互援引的关系，且联系方式时而精确、时而混沌，令人难以琢磨。

此外，这些交易所依赖的基础设备，都建立在持续发展了数十年的技术之上。这样一来，整个系统就成了“新”与“旧”的组合：在这个系统中，古老的依靠人工喊价的实物交易与通过光纤传输完成的电子交易共存。我们在构建能够实现高效交易的计算机程序时，不仅要考虑日新月异的计算机科学、繁杂的金融工具和深不可测的法律法规等，还要深入了解物理学。因为“光在不同材料中的传输速度”这一点在交易中至关重要。因此，我们说，地球上没有人能完全理解金融世界中相互关联的所有系统，甚至没有人能完全理解其中任何一个系统。

当然，在很多情况下，系统的使用者确实只需要很好地理解系统的一小部分，甚至仅是最表层的一部分就可以了。在一家金融公司里，程序员可能只需要知道如何维护交易系统，而不需要了解计算机在物理层面上的基础设置；有的人可能只会关注某个特定的软件，这些软件可以

对公司外部的消息进行过滤，然后将部分信息传输到公司内部进行运算，至于其他大部分信息，只作大致了解即可；为该公司工作的律师则需要了解与各种交易有关的法律条文，但不需要知道软件、服务器或光纤的任何细节。在这里，抽象方法给我们带来了很大的便利。

在大多数情况下，对某个事物“足够”了解，看起来已经很不错了，[28]但是，当我们构建出越来越复杂的系统之后，在系统及其子系统运行的不同层级之间就会越来越频繁地出现跨界交互现象。尤其是随着事物之间的相互联系不断加强，我们愈加难以判断原有的那些粗略的、不完整的理解是不是仍然够用。在纠缠中，事物会在不同的抽象层级上相互碰撞，以各种令人无法想象的方式进行交互。在充满交互的网络中，常常会出现被复杂性科学反复提及的“涌现”（emergence），也就是某种层次上的交互最终导致其他层次上的交互出现了预料之外的现象。涌现，在所有类型的复杂系统中都很常见。例如，昆虫的集体飞行便是一种涌现行为。另外，在金融系统中，涌现也很常见。金融系统的运行所涉及的因素多种多样，上至全球范围内的算法交互，下至每条网络线的传输速度。要想真正搞清楚哪些细节应该被抽象化，是一个太过繁杂的问题。

当系统内部深处的某些微小细节像“微型造物主”一样崛起，并开始破坏技术系统的其他组成部分时，我们就不能只对系统进行局部理解了。当系统处于纠缠之中时，其内部各部分之间的交互会陷入混乱，以往帮助我们管理复杂性的等级结构和抽象方法，都会迅速地分崩离析。

那么，在可见的未来中，人类是否有希望找到摆脱这种混乱的办法，并让世界重新回到可管理的状态呢？或者，我们注定要带着深切的、无法言喻的恐惧感来面对这些不断增殖的系统？

在大多数人看来，不完全了解那些技术系统，不知道所在城市的基础设施细节，不理解苹果手机的硬件如何验证指纹，不清楚法律法规如何促进国际贸易，其实并没有什么关系。人们简单地认为，对于那些复

杂的系统，只需要明白如何使用即可，它们的具体机制是什么则无关紧要。不过，对于一个新工具的工作原理，有人搞不懂是一回事，所有人都搞不懂又是另一回事。许多人还在自欺欺人地以为，专家们终会把我们从这种庞大的复杂性中拯救出来，因为他们理解我们所不理解的东西。但是事实并非如此，专家也不一定理解，依赖专家的时代早就一去不复返了。

我们过去所采用的用以理解这些系统的思维方式，也就是找到“挑战者号”航天飞机失事原因的那种思维方式，现在已经完全失效了。这很令人绝望。纠缠的世界并非远在天边，而是近在眼前。每个人都需要用新的思维方式去理解技术，甚至是那些我们轻松地将理解外包给专家的技术。

虽然对人类而言纠缠时代的到来是一场严峻的挑战，但我仍然满怀希望，并坚信：我们定能学会如何处理这些系统，至少在某种程度上。

但是，要想真正理解这个由人类一手创造的时代，我们就需要先退而求其次，将迫使我们陷入复杂性、阻碍我们理解复杂性的各种因素找出来。


技术系统变得越来越复杂的主要原因是“吸积”和“交互”。随着时间的推移，系统中不断加入更多的组成部分，部分之间也增加了越来越多的连接。“必须处理的例外情况”和“普遍的稀有事物”也让技术系统变得愈加复杂。

**要想使用互联网，我们就必须忍受或是间接地忍受杂乱无章、东拼西凑的网络世界。真是一团糟！互联网的第一步是在20世纪60年代迈出的。[1]当时，有人创造出了一个巧妙的设计，使人们能够在不同的地方通过计算机传递信息包。于是，小网络被相互关联起来，构成了大网络。同时，为了高效地传输信息，人们还开发出了各种各样的协议。**

**到了今天，互联网的应用已与其最初出现时大不相同，以安全问题为例。互联网原本是研究者以沟通为目的开发出来的系统，对于要求高**

效和安全的大规模商业交易而言，并不理想。为了弥补这种缺陷，促进商业交易，人们在原有的互联网基础设置上开发出了很多不同的机制，包括加密和解密信息的方法，以数字方式转移资金的规则，等等。令人高兴的是，这个系统确实变得有用了。然而，在网站的用户界面背后，其实潜藏着一个奇怪且复杂的结构。有时候，作为用户的我们也能直接窥见些许混乱的迹象，例如，网页上弹出的安全证书警告。是的，很多事物都是这样，有用，但是远远称不上优雅。

与此类似，开发网站所用的HTML语言，在设计之初，也并非是为了服务诸如谷歌在线办公软件（Google Docs）这类基于全球广域网（Web）的交互式应用程序。现在，这类应用程序已经投入运行，但人们仍然在付出代价：我们必须在一个简单的系统上建造一栋非常宏伟的巴洛克式建筑。倘若你想一瞥这种复杂性的一角，只需查看一下谷歌主页的源代码就足够了。虽然我们在浏览器中看到的网页既简洁又优雅，但是潜藏在这个表象下的东西却数不胜数。我上一次查看谷歌主页的源代码时，其字符数已超过了10万个，如果完整地打印出来，将超过50页纸！[2]

再比如电子邮件。从表面上看，这是一个相对简单的应用程序，已经发展了好几十年。在其古老的原始结构的基础上，消息线程等各种新功能层出不穷。不过，网络杂志《页岩》（Slate）的互动编辑克里斯·柯克（Chris Kirk）在尝试构建自己的电子邮件客户端程序之后指出：“虽然电子邮件的软件时有创新，但这些创新都建立在过时的系统之上，如同狡猾的平衡术，有时甚至相当随意，比如，将电子邮件恢复为最初形式，或是将它改头换面。”[3]

在计算机学和工程学中，有一个术语kluge，指的是拼凑起来的系统，也就是将许多不同的东西混合在一起，以求解决问题的系统。这种系统肯定是不精致、不优雅的，而且很多时候庞杂得毫无必要。虽然这种拼凑起来的系统是有效的，但远远称不上完美。有些东西的第一代设

计可能是相当优雅的，但是随着时间的推移，它们的结构变得越来越复杂，最终变成了杂乱的鲁布·戈德堡（Rube Goldberg）式[2]的应急之物。

包括互联网在内，在每一个技术领域中，都存在着这类拼凑起来的系统，例如交通网络和医疗设施。以家庭娱乐系统为例，它们虽然有用，但需要同时使用好几个遥控器，以及一大团乱麻般的电线、信号线和数据线。

美国的法律体系也是个拼凑起来的系统。这是一个为了达到特定目的而被构建出来的技术系统，远远称不上优雅。就像计算机代码是操作软件的书面描述一样，法律法规本质上也是技术代码的书面体现。[4]

毫无疑问，《美利坚合众国宪法》是一部非常优雅的文件，只用了寥寥数页，就为代议制民主奠定了坚实的基础。当然，宪法的确立并不是故事的结局。对联邦法律具有指导意义的《美国法典》是在宪法的框架内发展起来的。这些法律致力于阐明宪法的一般原则，以及对各种具体情况的处理准则和方法。例如，《美利坚合众国宪法》只用一句话规定了国会有权建立公共邮政服务机构，而在《美国法典》中，有关这个政府职能的阐述多达500余页。[5]此外，美国联邦邮政法规还规定了从邮政局的职位设置到邮政资费等各方面的所有细节。总而言之，《美国法典》比《美利坚合众国宪法》要复杂得多。[6]事实上，《美国法典》的规模和互联性仍在不断增加，时至今日，其总字数已经超过了2 200万，内部各章节之间的关联点也已超过了8万个。

其实，无论在什么地方，我们都可以观察到，随着时间的推移，各种系统的复杂性都出现了大规模增长。我们还发现，一般来说，当一个复杂的系统庞大到一定程度时，无论其具体形式如何，都会变成一个拼凑起来的系统。莱特兄弟于1903年制造的飞机是简约主义的杰作[7]，只有很少的几个部件，载人后总重量仅为340千克；而到了今天，制造一架波音747-400飞机[8]需要用掉67 000千克铝材，600万个独立部件

和275千米管线[9]。这是个普遍现象，在过去的200年里，我们制造出来的机器所包含的零件数量一直在大幅增加。

那么，对于影响着我们生活方方面面的，现代技术系统中的软件来说，情况又如何呢？衡量软件复杂性的常用指标之一是程序代码的行数。据估计，微软操作系统的源代码行数近10年增长了10倍。[10]Photoshop的源代码行数在过去20年里爆炸式增长，几乎是1990年的40倍。[11]

在电话通讯系统中，类似的情况同样存在，随之而来的还有巨大的复杂性。20世纪20年代，美国的电话通讯系统已经拥有了大约480万千米的收费线路和大约1 700万部电话。[12]要知道，就当时而言，电话才刚刚问世几十年；而时至今日，相关的技术生态系统已遍布全美。

所有这些系统都是为了实现某个特定功能，由一代又一代的专家设计、构建出来的。有人可能会认为，如果这些系统的设计是合理的，那么它们理应合乎逻辑、优雅，甚至简洁，也理应易于说明、易于修复。然而，尽管我们已尽了最大努力，但技术系统还是变得越来越复杂、越来越庞大。这绝非偶然，技术发展过程中某些固有的力量，使我们在“复杂性”中越陷越深。和万有引力之类的物理定律截然不同，这些力量强大到能让系统变得越来越复杂。不管时代如何变迁，它们总能压制住人们对简单的渴求，以致在人们心中，它们已如物理规律般不可抗拒。可是，为什么会这样呢？

在本章中，我将详细分析一些会使系统变得日益复杂的因素。从表面上看，这些因素完全合理，它们所引发的每一个变化都能使技术更加适应不断变化的环境，要么有助于系统继续在新环境中正常运行，要么增加了系统的实用性。然而，这些因素最终会使原本优雅的解决方案变成杂乱无章的、拼凑起来的系统。无论付出多大努力也无法避免这样的结果，无法阻止技术复杂性的不断增长。最终，我们生活的方方面面都会受到影响。

各种技术系统皆会随着时间的推移而越来越复杂，究其首要原因，也是最显而易见的原因是系统内部存在着双重力量：吸积（accretion）和交互，也就是说，随着时间的推移，系统的组成部分越来越多，同时组成部分之间的关联也越来越多。

## 原因1：吸积

在2000年1月1日到来前的几年里，许多工程师都在研究如何解决“千年虫”问题。简而言之，若一个软件在存储年份时使用的是两位数，而不是四位数，那么当2000年到来时，这个软件就会“认为”那是1900年，从而引发诸多问题。恐怕没有任何人、任何机构会比美国联邦航空管理局（FAA）更担心“千年虫”问题了。如果空中交通管制系统在新千年到来时出现故障，那将会导致巨大的灾难。因此，美国联邦航空管理局检查了计算机系统，并对系统进行了测试：如果系统认为那是1900年，会出现什么情况？[13]

在测试过程中，他们发现交通管制系统中的IBM 3083计算机问题特别棘手。作为技术人员代表，美国联邦航空管理局的工会主席指出主要问题在于：“IBM公司只有两个人知道这种型号的计算机的微代码，但是他们都退休了。”[14]但真正的原因是，IBM 3083属于大型机，从20世纪80年代开始发售，而且所用的系统软件早在发售前几年就已投入使用。这就意味着，到了20世纪90年代后期，这个为全美的飞机制定航线的计算机系统所使用的代码几乎已无人可识。

这还算不上令人震惊。一直以来，许多大型系统的基础都是较小和较陈旧的系统。只要这些系统能够继续平稳地运行下去，就不会有人在意那些旧东西上面到底堆了多少新东西，系统中到底累积了多少次第加入的片段。据一位消息人士透露，直到2007年，美国国税局所使用的报税系统依旧是20世纪60年代早期，也就是肯尼迪政府时期开发的系统。[15]美国国税局所使用的另一个系统则始建于20世纪70年代，并于

1985年进行了大修。与此类似，美国的航天飞机在执行最后一次航天任务时，[16]所用的平台是由5台IBM计算机组成的，其计算能力甚至还比不上今天的一部普通智能手机。然而，这些软件和技术仍然在使用着。

1975年出版的《人月神话》（The Mythical Man-Month）一书中，计算机科学家小弗雷德里克·布鲁克斯（Frederick P. Brooks Jr.）讨论了有关软件设计和项目编程的管理问题。在这本书中，他引用了业内的一句俗语：“每次加一点，每次加一点，最后就有了一大堆。”[17]每一个独立设计，无论是为了修复，还是用来提供新功能，看上去都不过是一次独立的选择，而且都很合理：要么解决了问题，要么为用户创造了新的令人兴奋的功能。然而，日积月累下，它们终会变成“一大堆”。从交通运输业到能源业，再到农业，我们都可以从中清楚地看到，但凡是大型的技术系统，就定会发生这种情况。

举例来说，一大堆石头不一定会成为问题，即使它们看上去可能很难处理、过于零乱，但却不一定深奥难解。真正的问题是，当我们创造的“一大堆”引发出意料之外的状况时，我们就会遭遇“雪崩”。不幸的是，我们不断地往技术系统中加入一个又一个片段，“雪崩”在所难免。“一大堆”不仅变得更大，而且变得更难琢磨了。

我记得我最早是在一些讨论“行星系如何形成”的文章中看到“吸积”这个术语的。行星系由一团旋转的尘埃和气体凝聚而成，这种星星点点的累积过程，就是“吸积”的过程。[18]这个用来描述行星系如何形成的概念由来已久，而在技术的增长过程中也是类似的吸积作用。

吸积过程的结果之一就是形成了人们常说的遗留代码（legacy code）或遗留系统（legacy systems），即过时的机器和技术，也就是开发出来之后使用至今的机器和技术，譬如美国国税局所使用的报税系统。这种老旧的系统并不罕见，甚至可以说相当常见。[19]它们经历了多年的吸积过程，成了拼凑起来的系统，从科学模型到城市基础设施，

几乎无处不在。例如，在城市的排水系统中，既有服役超过百年的旧管道，也有刚刚埋设好的新管道。

就计算机领域而言，很多技术系统都依赖于已停产的旧型号计算机系统，而且程序代码也是用早已退役的编程语言所写。例如，许多科学软件现在都成了遗留系统，它们一般都是用Fortran语言编写的，那是一种功能强大但早已过时的编程语言。随着技术的高速发展，如今我们再来看Fortran语言编写的计算机代码，就好像是在看中古时代的英语。

在这里，不妨引用《全球概览》（Whole Earth Catalog）的创始人斯图尔特·布兰德（Stewart Brand）在《万年钟传奇》（The Clock of the Long Now）一书中的说法：“通常，这些已经过时的遗留系统，在过去的许多年里，一直都扮演着十分重要的角色，若是替换掉它们，必定会‘伤筋动骨’，恐怕没有人能够承担这样的后果。此外，它们也是无法被完全修复的，一是因为问题过于复杂，二是因为没有人完全了解整个系统。”

那么，当我们面对一个仍在缓慢增大、小故障不断的遗留系统时，又该怎么办呢？我们只能小心翼翼地对待它，因为设计它的人可能早就“杳无音讯”了。这个遗留系统可能已经完全嵌入了其他系统，彻底移除它的后果，可能远比容忍它的小故障更加糟糕。这种系统实在难以处理，有时甚至会被称为“恐怖爬行兽”（crawling horrors），也就是美国恐怖、科幻小说作家霍华德·菲利普斯·洛夫克拉夫特（H. P. Lovecraft）小说中那种无法形容的怪物。[20]

司法系统条文，也有类似的情况。在法律体系中，随着时间的推移，法律条文会被修正或修改。人们会根据不断变化的环境，对法律条文做出调整，但总会留下一些数十年前制定的法律条文。例如，互联网流量管理的相关法规就源于1934年通过的一项法律。随着时间的推移，法律体系也在不断地吸积，最终成了一个拼凑起来的系统。它可以

发挥作用，但是远远谈不上优雅。

以税法为例。事实上，美国的税法早已复杂不堪了，立法者也早就承认了这个事实。税表的使用说明已经从1940年的两页，增加到了2013年的200多页。[21]鉴于此，如果你在申报纳税时，因为法律法规太过复杂而出了错，那么最高法院会裁定你无罪，因为你在错误报税这件事上并没有“主观故意”。[22]尽管如此，就法律体系本身而言，在现有税法上修修补补，要比重新制定更容易，也更有效。

另外，我们也要考虑一下诸如美国环境保护署之类的政府部门和机构所颁布的法规的总体增长趋势。事实上，只要看一看《美国联邦法规》的总页数就足够了。[23]《美国联邦法规》收录了各行政机构颁布的各种法规，在过去的50年里，其页数从不到25 000页增加到了超过165 000页。

我们还观察到，行政人员和行政机构的数量也出现了类似的增长趋势。20世纪50年代，《经济学人》杂志中有一篇文章提到了帕金森定律（Parkinson's Law），并定量地描述了行政人员数量的增长规律。[24]虽然这篇文章的观点还不太完善，但帕金森定律毕竟有数据的支持，其结论到今天仍基本成立：政府部门的行政人员数量以每年5%至6%的速度在增长。毫无疑问，随着行政机构的规模越来越大，机构的管理问题只会越来越复杂。

事实上，软件界已经将吸积和积累奉为普适规则。[25]就发展而言，软件系统的规模势必会与日俱增，除非有人积极地尝试简化它们。

那么，为什么我们不能对复杂的系统进行定期清理并从头开始呢？这和实际操作有关。例如，软件未能按时重写完，索性就在推出新版本之前先发布一个补丁。[26]我在想，只要愿意花上很多年的时间，那么微软内部的任何一位软件专家都可以重写Word的全部代码。当时间、精力和金钱都有限，且不得不进行权衡时，[27]我们通常会选择对系统进行修改，让它“足够好”就行了。这意味着，我们需要不断对系统进行调

试和修正，就像立法者对美国法律体系所做的那样，也就是说，我们需要在以往的基础上加上一层又一层的东西。我们的城市拥有一个多世纪之前埋设的燃气管道[28]、20世纪30年代建成的运输网络[29]，以及废弃的地铁站。它们都隐藏在城市的地下。

不过，更多的时候，我们之所以放弃从头开始的想法，是因为那样做不仅太困难，而且太危险。没有人能够完全理解一个系统所依赖的所有旧的组成部分的全部作用，所以重新设计一个未经检验的系统不仅是愚蠢的行为，更是危险的行为。试想一下，一个几十年前设计好的、非常复杂的银行软件系统正慢慢适应着各种先进的技术，无论是新型的计算机，还是新的操作系统，或是无处不在的互联网。虽然这个系统的核心基础并不适用当今时代，但它们已经嵌入得太深，以致无法删除。总而言之，我们必须接受这样一个普遍规则，那就是：无论何种技术系统，终将变得日益复杂。[30]

但是，当仔细观察技术系统中的遗留代码时，无论是在一个软件中，还是在一个法律体系中，我们都会发现，真正的复杂性绝不仅仅只体现在日益扩大的系统规模上。毕竟，只有和另一个因素结合起来，吸积才能使技术系统变得复杂，这个因素就是交互。

## 原因2：交互

现在，几乎所有的学生都会在师长的引导下，学习一些编程技术。是的，不管是教育工作者还是技术专家，都告诉我们计算机编程就是未来。计算机操纵了周遭的一切，比如汽车和微波炉。专家们当然没有说错。不仅如此，编程还可以为你带来一种结构化的思维模式，以及提示你关注某种技术系统的实际功能。当我说起某个新的应用程序能够做某件事时，如果你参与了编程，哪怕只是一点点，你也会比一般人更清楚我的说法是否合理。

如果你以前写过代码，那么你应该清楚计算机程序是如何抵制简化工作的。随着计算机程序变得越来越庞大、越来越复杂，代码也变得越来越复杂，而且绝大部分代码的聚合方式都是令人费解的。为了确保大型程序仍可控，我们开发出了各种各样的技术方法，例如版本控制、错误跟踪，以及跨团队沟通工具等，但是这些方法通常都只是“战斗失败”后的亡羊补牢之术。不仅软件代码本身在不断吸积，每个组成部分也在越来越频繁地与其他组成部分进行交互。与存在于“真空”中的全新项目有所不同，任何一个计算机程序都是一个大规模的互联系统：不仅会作用于自身，还会与其他程序相互作用。我们在旧代码中一次又一次地加入新代码，并以出其不意的新方式去使用它们，以此将各个层次拼接在一起。

交互过程，包括许多意外交互在内，有时会因编程语言本身的原因而加剧。即使是刚入门的程序员也知道，GOTO语句会带来麻烦。在BASIC编程语言中，通过设置GOTO语句，可以让程序轻松地从某一行跳转到另一行。换句话说，如果代码中包含了GOTO语句，那么程序就可以轻松地从某一行指向另一行，从代码中的一个点跳转到另一个点。如此容易便能实现跳转，难怪有人会说，对于那些想要对自己的计算思维进行测试的文科生而言，GOTO语句无疑是“天赐良品”。[31]

在一个小程序中，使用GOTO语句实现跳转，既容易又无伤大雅。但是，随着程序变得越来越大，GOTO语句最终会将代码绑定到某些庞杂的节点上，但即使是最熟练的程序员也无法解开这些庞杂的节点。最终，你得到的将是所谓的“意大利面条”式的代码，因为所有东西都纠缠在了一起，既难解开，又难理解。在这种情况下，要想搞清楚程序指令执行的次序几乎是不可能的，这也是为什么这样的计算机程序特别容易出现令人意想不到和无法理解的行为。

此类简化命令即使只是在独立的、较小的环境下发挥作用，也有可能通过某种方式脱离当初设定的目标。如果使用的频率和方式超出了预

设范围，系统及其组成部分就会出现诸多问题，而这些问题比当初设想的要庞杂得多。GOTO语句就是一个突出的例子，它从一个美妙的超快捷工具变成了既不优雅，又实为“有害”的东西。[32]

为了更好地理解并强化系统秩序，专家们想出了很多方法，包括使用更加复杂的计算机语言。[33]在大多数由专家构建的新系统中，意大利面条式的代码已成为过去式。然而，由于互联的易发性和各种层次上的不断吸积，交互作用正在持续增多。于是，高度互联的系统动态，即信息的流动和各部分之间的交互，也就变得异常复杂和不可预测。仍以丰田汽车软件为例，因为存在交互，丰田汽车软件中的大量代码已无法被测试了。[34]

在我们构建的其他类型的技术系统中，互联性同样在不断升级。例如，在法律体系中，由于每项新的法律法规都与以前的法律法规相互关联，因此我们很难预测单项法律条文的效果。

菲利普·K.霍华德曾仔细分析过贝永大桥（Bayonne Bridge）这个案例。[35]贝永大桥是连接纽约州和新泽西州的主要通道之一。由于这座百年大桥的桥梁实在太低，以致那些前往纽瓦克港的现代集装箱船无法顺利通过。纽瓦克港是一个重要的商业中心。那么应该怎么办呢？在人们提出的各种解决方案中，有一个方案是这样的：对这座桥进行改造，适度提高桥梁高度。这应该是成本最低的方案了，也是在2009年脱颖而出的方案。但是，改造工程拖了很多年都未能启动，因为人们难以应对吸积和交互的综合作用。与大桥改造工程有关的规章制度总共涉及19个政府部门的47份许可文件，从环境影响评估报告到历史影响评估报告，不一而足。[36]其他地方也出现过诸多类似情况，一些公共项目需要10年左右的时间才会获得批准，因为相关的规则和流程冗长繁多。[37]正如霍华德所说，这种情况在很多时候甚至是致命的，比如，老化腐朽的基础设施如果未能得到及时修缮，就有可能夺走许多人的生命。

研究员迈克尔·曼德尔（Michael Mandel）和黛安娜·卡鲁（Diana Carew）就职于位于华盛顿特区的进步政策研究所（Progressive Policy Institute）总部。他们将规则体系的增长称为“监管积累”（regulatory accumulation），即随着时间的推移，规则会变得越来越多。[38]换句话说，每一条法律法规都是合理的，但当它们被放到一起时，就有可能会因为相互作用而变得异常“软弱”，甚至可能以令人惊讶和意想不到的方式产生冲突。

我们不仅越来越多地将某项技术的各个组成部分关联起来，而且还越来越多地将不同的软件和技术关联起来。后者是一种高阶互联模式，也就是互操作性（interoperability）。[39]

让各项技术互通，也就是让不同系统进行交互，相互传递信息通常是一件好事。例如，因特网之所以拥有如此强大的功能，就是因为其连接的机器数量极其庞大，而且可以在无数机器之间传递信息。当你问Siri“世界总人口是多少”时，你的苹果手机会通过Wolfram Alpha服务获取到答案，然后回答你；当你使用谷歌地图时，它会告诉你利用“优步”去往目的地可能要花多少钱。这些都是互操作性的实例。但是不要忘记，让不同系统互通的同时，我们不得不去面对陷入复杂世界的巨大风险。我们现在不仅建成了互联网络，比如由不同计算机和设备组成的互联网，而且还建成了拥有众多子系统的大型互联系统。

除了互操作性之外，不同类型的技术之间还会产生相互依赖性，例如互联网与电网之间的相互依赖性。[40]研究者在研究了多种类型的系统，并了解了它们的优缺点后指出，某些系统在多种条件下均可能会出现故障或崩溃。例如，某个规模相当小的电网出现了故障，继而引发了无法收拾的级联效应。对于这种风险，有一种观点是：将技术系统之间的相互联系切断。然而，这种想法的可操作性几乎为零。互联系统的构建成本其实很低：在当今这个充满互操作性的时代，工程师和设计人员都在有意为各个系统创建接口，因此，不同的系统可以很容易被关联起

来。

我们在构建新事物时，通常都会在故障成本和构建成本之间进行权衡。我们需要知道，如果出了故障，失败的成本会有多大。[41]如果Word崩溃了，那么尚未保存下来的东西将会丢失。尽管没有人希望看到这样的结果，但这个故障的成本的确是相对较低的。如果电网出了故障，并导致美国很多地区停电，那么故障的成本就极其高昂了。例如，在2003年，美国东北部的大停电对5　000万人的生活和工作造成了影响，并导致11人丧生，直接损失估计高达60亿美元。[42]

每一项故障成本都应该拿来与系统的构建成本进行比较。纵观历史，我们所构建的系统越重要，构建成本就越高。例如，构建银行系统的基础设施所耗费的资源，比编写一个聊天程序多得多。因此，我们必须确保那些昂贵的系统不易发生故障，而这又意味着需要增加构建成本。换句话说，在极高的构建成本面前，通过大量检查和测试来降低故障成本的做法，变得至关重要。

在一个相当长的时期内，这种方法一直行之有效。因为对构建成本的重视程度超过了对故障成本的重视程度，所以我们所依赖的所有重要的社会保障系统，都是花费大量资源构建起来的。[43]然而，现在事情已经发生了变化。出于各种原因，例如，我们可以找到现成的工具和组件，同时“云”上面也存有很多可用资源，所以构建成本已大幅下降。创立技术公司已不再需要太多启动资金：你可以快速地设计和生产出复杂的工具，并通过市场进行测试，而你为此所付出的成本并不大。

与此同时，与互联有关的故障成本也已出现持续上升的趋势。尽管系统关联技术相当简单，而且成本很低，但是这种互联系统的故障成本却非常巨大。当我们把数字地图软件与出行指导软件关联到一起时，哪怕只是一个很小的错误也有可能导致一场灾难。例如，苹果地图在首次发布时就曾将超市错标为医院。

在当今时代，利用互联网信息人工合成微生物并不是一件不可思议

的事情，但正因如此，爆发生物灾难的风险也比以往任何一个时代都高得多。例如，已经有实验室通过使用电子邮购的生物原料合成出了脊髓灰质炎病毒。现在，有不少初创公司正在努力实现生物学实验的远程操作。不难想象，在这个日益自动化的世界中，软件合成的生物因子[45]完全有可能会在不经意间被释放。当构建成本持续暴跌，而故障成本直线上升时，我们便进入了一个无比复杂的技术领域。是时候停下来想一想了。

一般来说，随着系统内部及系统之间的交互增多，包括拥有子系统的大型系统在内的所有系统的复杂性都会增加。有人认为，互联性的不断提高，体现了技术的基本要求。[46]技术终究会产生交互和聚合，并在这种情况出现时进一步推动我们走向复杂化。

尽管从开始构建大型技术系统的第一天起，这些趋势就一直存在，但是近年来，它们变得愈发强大了。正如我在本书导论中所提到的那样，计算机科学家艾兹格·迪科斯彻对当下的大型系统，特别是计算机系统的激进新颖性所进行的分析发人深省。早在1988年，迪科斯彻就已指出，计算机程序的设计需要克服大量规模上的差异，当然，在计算机问世之前，没有人需要去处理这样的事情。[47]以智能导航系统为例，迪科斯彻很好地解释了其跨度极大的层级结构。从程序中的1个比特，到机器存储空间里的几百兆字节，的确是从非常小到非常大的跨越。这种跨越涉及近10亿次的跳转，极端的规模变化不但超乎想象，而且史无前例。事实上，这一切已经，并将继续走向极端化，因为日常应用技术的普通用户，现在已经熟悉了千兆或万兆这样的前缀，而这些前缀又意味着，我们要对庞大的规模差异负责，而这种庞大系统的边界已近乎天文意义上的边界了。

在过去的短短几十年间，大型系统已变得异常庞大且错综复杂，用迪科斯彻的话来说即是“概念层级的深度，绝非人类心智曾需面对的任何事物可比”。

不仅如此，即使我们有能力阻止系统的吸积和交互，也还需面对另一个会使系统变得日益复杂的因素，而那将是更加难以解决的问题。

## 原因3：必须处理的例外情况

假设你想创建一个日历程序。看上去很简单，是不是？从很多方面来看，的确如此。要想计算一年的天数，相对来说并不算困难。一般来说，一年有365天，遇到闰年就在2月底加上一天。怎样计算闰年？也不难。只需要看一下年份即可。如果年份可以被4整除，但无法被100整除，或是能被400整除，那么这年便是闰年。

也许你还希望这个应用程序能够处理时区问题。这应该也不算难。只需利用全球定位系统（GPS）中的地理坐标来确定所在位置的时区即可。你还可以创建一个列表，将各州与时区对应起来。当然，时区并不是沿着州界划分的。现在你需要用更具体的、分辨率更高的信息来处理小区域。此外，你还不能忘记，亚利桑那州的大部分地区都是不使用夏令时的，它把自己置于一个“特殊的时空”当中了。

是不是还想让日历程序包含节假日的相关信息？当然，节假日通常都有明确的时间安排，至少大部分节假日都是如此，把它们加上去应该不会太困难。感恩节是11月的第4个周四，美国退伍军人节被定在每年的11月11日。那么逾越节（Passover）呢？看来，我们还需要将这个日历程序与另一个基于希伯来历的日历程序整合到一起。因为逾越节是从希伯来历一月的第15夜伊始的。因此，我们需要远超预期的更多信息。

那么，你是否还希望这个日历程序包含其他时间段，并且在过去时段和未来时段上都表现得准确无误呢？如果我们回到19世纪，在那个时代，由于标准化时区制尚未被推行，所以各个城镇都有自己的时间制度，而这些信息都需要“硬编码”（hard-coded）[3]到我们的应用程

序中。与此类似，在过去的几个世纪里，虽然全球许多国家和地区都已放弃使用罗马儒略历（Julian calendar），转而使用格里高利历（Gregorian calendar），但是各国以及各地区所采用的具体时间制度并不全然相同。例如，俄罗斯“十月革命”的纪念日之所以是在11月，是因为在革命发生时，俄罗斯仍在使用罗马儒略历，其日期与西方许多地区所使用的格里高利历相差一周多，而“十月革命”发生在儒略历的10月底。如果希望日历程序准确且详尽，那么就应该将此类信息也植入进去。

这个过程还将持续下去。

以这种方式构建起来的系统最终会变得复杂无比，因为它所要反映的事物，本身就是复杂的。[48]通过一个简单的模型来处理绝大多数复杂性，是相对直接的方法。比如，我们知道了一年有365天或366天，就可以通过简单的运算来确定某一年到底有多少天。但是，如果你对准确性有要求，无论是想确保永远不会错过任何一个约会，还是想构建一辆既不会迷路也不会撞伤人的自动驾驶汽车，事情就会变得非常复杂。[49]

这种复杂的情况就是必须处理的例外情况，也就是所谓的“边界情况”（edge case），若不处理，技术系统就会出现漏洞。[50]边界情况各种各样，从闰年问题，到如何编写数据库软件来处理特殊的人名，比如人名中带有特殊符号的情况。我们不能说边界情况是普遍现象，但它们确实经常出现，所以我们必须加以识别和管控。但是与此同时，技术的简单性也就渐渐消失了。边界情况使技术变得复杂了。这一点在科学模型中尤为突显，科学模型也是一种技术，也会随着时间的推移而发生变化。接下来，我们就以社会科学中的语言学为例展开讨论。

## 原因4：普遍的稀有事物

在中学时期，英文老师教会了我语法，因此我很早就知道，不规则

动词to　be的两个单词是必须连在一起用的；我还记住了很多介词，并学会了造句。在那个时候，根据语法规则来分解句子，将句子的修饰成分去除，剥离出它的逻辑骨架，是一件很有趣的事情。你可以将语言简化到原子量级，如名词、动词和形容词，然后再来看它们是如何关联在一起的。

尽管语言无法用方程式来表达，但是语法确实拥有一种独特的、有秩序的美感。然而，构建一个语言处理系统并不是一件容易的事。任何一种语言都有很多习惯性表达，而且其内涵往往比我们想象的要丰富得多、“狡猾”得多。因为语言具有非正式性，所以使用者在面对作为规则集合的语法时，多半只会深表认同，而不会严格遵从。所有这些都属于同一类边界情况，它使下面这个简单规则无法成立：每个句子都必定是“主语－谓语－直接宾语”这种结构的变形。为了更好地理解语言中的边界情况，我们现在来讨论一下所谓的罕用语（hapax legomena）。罕用语可谓“普遍的稀有事物”（common rarities）。

请问你以前用过snowcrie这个单词吗？我想你应该没有用过。事实上，snowcrie这个词是无意义的。据我所知，它可能是一个错词。根据《牛津英语词典》的解释，snowcrie这个词曾经出现在1402年的一首诗中：“Not in Goddis gospel, but in Sathanas pistile, wher of sorowe and of snowcrie noon is to seken.”。有学者认为它应该是一个错词，[51]正确的那个词可能是sorcerie，意为巫师。

不管有没有意义，snowcrie这个词就是所谓的罕用语，或者说“只用过一次的词”。这个词在《牛津英语词典》的语料库中只出现过一次。语料库是大量的、通常是完整文本的合集，例如某种语言的全部文本，或某个时期的所有文本。《牛津英语词典》的语料库即是编写者可以使用的所有英语文本。不过，语料库的文本体量并不一定非常大。在莎士比亚文集这个语料库，也就是莎士比亚的全部著作中，经常

会碰到一些罕用语，比如honorificabilitudinitatibus，含义可能就是“荣誉”（of honor）。

当一个语料库拥有某种语言的全部，或近乎全部的文本时，罕用语就会变得让人头疼，比如《希伯来圣经》中的希伯来语，人们对它们的意义知之甚少。但是，罕用语并不是离奇的统计错误。它们不仅比想象中更加普遍，而且与语言学中特定的数学规则有关。语言中不同词汇的使用频率可以用幂律（power law）[4]中的长尾分布来进行描述。[52]长尾分布与用来描述人类身高的钟形曲线（bell curve）[5]，即正态分布，有所不同。在长尾分布中，有一些值会延伸到更加深远的区域，以便容纳普通词汇，比如the，或一些极其罕见的词汇，比如flother。

一般来说，语料库中有近一半的单词都只出现过一次，也就是说，有一半的单词都属于罕用语。这些词就是长尾中“长”的部分。[53]因此，虽然你遇到某个特定罕用语的概率很低，但是你遇到这类词的概率却相当高。在这里，我们不妨用看电影来做下类比。相信并没有太多人看过那部大名鼎鼎的经典电影《天生爱神》（The Adventures of Buckaroo Banzai Across the 8th Dimension），但是看过至少一部经典科幻电影的人却大有人在。

因此，作为一个整体类别，罕用语是非常重要的。它们深深地渗透在我们的语言之中。当我们试图编写一个计算机程序来模拟语言时，可能会将罕用语，或罕见的语法结构抽象为异常值。但是，作为一个类别，而非一个单词，罕用语在语言中所占的比例其实是相当大的。将它们抽象化会导致模型的严重缺失，从而使程序变得不完整。为了避免“遗漏”，[54]我们需要建构出可以处理例外情况和边界情况的复杂模型。在这个问题上，谷歌公司研发主管彼得·诺维格（Peter Norvig）的话可谓一语中的：“形成一种语言的，并不是那种可以用几个参数来代表的永恒的理想模型，而是复杂过程中的偶然结果。”[55]

因此，计算机语言学家应该考虑边界情况，并尝试着针对这个复杂

系统构建一个稳健的、丰富的技术模型。在这里，复杂系统指的就是语言。那么，他们最终会得到什么呢？毫无疑问，他们将会得到一个复杂的技术系统。

和语言有关的计算机模型必定具有复杂性。要说明这一点，只需举一个例子就够了：计算机是如何将一种语言翻译成另一种语言的。关于计算机的翻译功能，有一个流传已久但未经证实的故事。[56]在冷战期间，科学家们就已开始研究英俄语互译的运算方法了。在测试计算机的翻译程序时，他们选择了一个含义相当微妙的句子“The spirit is willing, but the flesh is weak.”（灵固有所愿，肉却软弱不堪）。他们通过计算机将这句话翻译成俄语，然后再次翻译成英语，最终得到的是“The whiskey is strong, but the meat is terrible.”（威士忌很有劲，但是肉却很难吃）。

显然，通过运算来实现机器翻译功能并非易事。谷歌翻译虽然有趣，但结果却可能不够准确。不过，专家在这方面已经取得了很大的进步。

那么，机器翻译专家使用的是哪些技术呢？[57]早期的一种方法是利用语言的结构化语法，进行模型搭建。计算机语言学家将每种语言的属性硬编码为软件，然后让计算机根据语法规则进行翻译。这种方法可以处理相对简单的句子，但无法应对日常语言的多样性。例如，一个用来处理“直接不定式”的规则，不一定能够处理“分离不定式”，即无法处理“To boldly go where no one has gone before.”（勇敢地进入前人未曾涉足之地）这样的句子。另外，不定式的用法还具有一定的地域性，比如，匹兹堡人很喜欢省略掉to be，直接说“The car needs washed.”（这车该洗了）。很显然，面对这种形式灵活的“方言”，语法规则将束手无策。

事实上，依赖上述语法模型的机器翻译程序是不可能给出准确结果的。语法规则看上去既优雅又简洁，但无法应对文本翻译过程中所需处

理的，复杂且古怪的语言现象。简而言之，边界情况实在太多了。为了填补这道鸿沟，机器翻译专家引入了机器学习领域的多种统计方法，他们让计算机先摄取大量已经翻译好的文本，然后基于一组算法翻译新文本。这样一来，计算机就不用理解句子的含义，也不必解析句子的语法结构了。举例来说，对于复数问题，我们不再需要创建复数的语法规则，规定将后缀“-s”加在单词末尾，就能够使之变成复数形式；我们只需让机器知道，“-s”这个后缀在99.9%的情况下意味着创建了一个复数形式的单词，而在剩余0.1%的情况下并非如此。[58]例如sheep和deer，它们的单复数形式相同；此外，还有一些单词的复数形式是不规则的，如men、feet和kine。对于语言系统中的其他例外情况，也都可以采用这种运算方法。

尽管摆脱混沌就能迎来秩序，但这不可能没有代价。最终得到的最有效的翻译程序肯定不是一个简单的模型，而会是一个拥有大量参数的庞大的计算机系统，若非如此，便无从处理数之不尽的边界情况和语言“异象”。正如谷歌翻译开发团队的成员所说，这类“基于数百万具体特征的模型，要比那些关注一般规则的精巧模型表现得更好”[59]。我们必须要对例外情况加以珍惜，绝不能随手丢弃。要知道，无论是例外情况，还是罕见情况，都包含了大量的信息。

这种机器学习技术利用的是概率和大量参数，而非原则性的规则。[60]这种尖端技术正越来越多地被应用于科学领域和其他诸多领域，从犯罪侦测到医学诊断，再到保险推销，等等。事实上，我们的审美品位也相当复杂。奈飞公司在向一个团队颁发“推荐引擎改进奖”时发现，该团队的解决方案是由各种统计技术拼凑而成的。这场比赛似乎表明，没有哪个简单的算法能够显著提高推荐的准确性；无论谁是获奖者，都需要使用更复杂的方法来捕捉和预测人们对电影的个性化需求和“怪异”品位。

这种现象其实在所有类型的技术当中皆有呈现。计算机科学家小弗

雷德里克·布鲁克斯明确指出：“软件的复杂性是软件的根本属性，而不是偶然属性。”[61]

即使是法律体系这个复杂系统，也会受到例外情况和边界情况的影响。很多人都认为，在合法与非法之间有一条明确的界限，但事实并非如此。恰恰相反，边界会随着时间的推移而不断伸缩、折叠，变得凹凸不平或纠缠不清，所以边界不可能是一条明确的界限。最终，法律体系看上去会是一个分形结构：无论你将一个小的图形放大多少倍，依然会发现很多的不均匀，依然有更多的细节需要进一步观测。[62]任何的一般性规则最终都必须应对例外情况，而后者又会裂解成更多的例外和规则，从而形成越来越复杂的分支结构。

对此，法学家杰克·巴尔金（Jack Balkin）在一篇题为《法律思想的结晶结构》（The Crystalline Structure of Legal Thought）的文章中阐释道：

> 我们可能会想，在关于疏忽的客观标准中，是否存在适用于儿童的例外规则；或者，是否存在适用于疯子、盲人，以及其他特殊人群的不同标准。这样的想法会引导我们开始进行接下来的规则选择，而且每个选择都会滋生出更多的法律含义分支。举例来说，假设我们遵循某个法律含义分支的发展轨迹，为儿童制定了一个例外规则，当然，现在这是一个多数规则。我们可能还会继续考虑到，当孩子从事成人活动时，是不是也需要一个例外规则，而现在，这通常也是一个多数规则。然后，我们可能会继续追问，在该规则的含义范围内，驾驶摩托车是否属于成人活动，如果是，那么驾驶小轮踏板摩托车是不是也属于成人活动？这个过程可能会一直持续下去，最终我们会得到一个关于规则选择的递减序列，但是事实上，复杂性和特异性都在不断递增……[63]

法学教授戴维·波斯特（David Post）和生物学家迈克尔·艾森

（Michael Eisen）也携手研究了这个问题。[64]虽然他们无法证明，任何一个法律陈述都能被进一步细分，毕竟这是一个“乌龟背上的世界”式的命题[6]，但是他们指出：“我们确实从来没有遇到过无法分解为子问题的法律问题。”波斯特和艾森还通过模型证明，某些类型的法律分支结构实际上具有分形结构的特性。在现实中，他们对诉讼案件中的法官意见进行了分析，并在引用的法律条文中也发现了指向分形结构的特征，从而验证了模型的结论。[65]由此看来，法律体系的分形复杂性很可能不只是一个令人回味的隐喻。

正如法律学者马克·弗勒德（Mark Flood）和奥利弗·古迪纳夫（Oliver Goodenough）所指出的那样：“好合同和好律师的价值，在很大程度上基于这样一种看似烦琐的规划：对于随时可能因出轨而覆灭的婚姻关系，要如何对一切可能的出轨方式未雨绸缪。”[66]换言之，法律体系的复杂性通常源于例外情况及其引发的“并发症”。

我们所观察的技术系统，无论是法律、软件、设备，还是科学模型，都会在例外情况和边界情况的驱动下，在吸积和交互的双重助力下，变得越来越复杂，越来越混乱。

## 越来越多的复杂系统

尽管我在上文中将那些导致系统越来越复杂的驱动力描述为不可抵挡的力量，但是我们还不至于束手无策。实际上，我们也进行了一些抗争。重要的是，在奋力抗争的过程中，我们看出了它们的“真正实力”，并洞晓了根除它们的难度。

以下策略似乎可以让技术系统变得有序且有逻辑。针对不同的系统，如果我们能够采用不同的方式去构建、设计、修改和重建的话，那么，对技术的控制或许真的可以实现。例如，我们可以尝试解耦某些系统，将它们拆解为更小的单位，以保证它们的相对简单性和可管理性。

拥有物理学背景的社会学家、微软研究院首席研究员邓肯·沃茨（Duncan Watts）指出，在金融领域，解决由复杂性导致的崩溃故障的方法之一，是直接消除耦合复杂性。[67]这就好比，如果某家公司的规模已经变得太过庞大，且预期其失败会造成级联式冲击，那么就必须对它进行拆分，或缩小其规模。

其他学者也讨论过类似的问题：怎样才能让一个大型系统的互操作性达到最佳水平。[68]这里所说的最佳水平是指，系统既能运行良好，又不会因高度的不可预测性而产生负面效应；也就是说，我们的目标应该是创造最佳水平的互操作性，而不是最大限度的互操作性。当然，这个目标不是那么容易就能实现的。想到是一回事，做到又是另一回事。实现这个目标的方法之一是坚持使用特定的设计原则，将可理解性和模块化内置于我们的设计中。

我在前文中已经阐述过，当一个系统具有高度的互联性时，我们便很难将其拆解，也就很难探究其内部发生的一切。但是，在大型系统中，确实可能存在这样的情况：某些部分之间的互联性远强于它们与其他部分之间的互联性。换句话说，系统中存在若干模块，而每个模块由若干“部分”紧密互联而成，并在一定程度上保持独立。我们常在生物学中见到这样的模块，它们皆包含了若干“行动一致”的“部分”。植物的线粒体和人体的心脏皆是如此。当然，这些模块通过别的身体器官和化学信号，以及其他方式，和系统其他部分保持着紧密的联系，所以我绝不建议人们进行心脏摘除手术。不过，这些模块又是相对独立的，即使不借助系统的整体功能，也可以被理解。

在技术系统中，我们同样可以观察到模块化现象。当一个软件拥有多个独立的功能或部分时；当你交替使用不同的应用程序来完成同一项工作时[7]；当你分析《美国联邦法规》中那些相对独立的部分时，你都可以看到模块化的身影。模块化遵循了抽象的原则，实质是通过将系统拆解为若干部分，从而在一定程度上实现对复杂性的管理。

然而，从单个模块入手去理解系统，或者在单个模块的基础上构建系统，也并非总能让我们如愿以偿。如果每个模块都是多向输入和多向输出的话，那么当它们关联在一起时，系统的行为很可能仍是令人难以理解或无法预测的。最终结果很可能是“组合爆炸”：由于潜在的、不同的相互作用太多，模块数量很快就会超出我们的处理能力。例如，假设系统中的每个模块都分别具有6个不同的输入路径和输出路径，同时这个系统拥有10个模块，那么将所有这些模块关联到一起的方法将比整个宇宙中的恒星还要多。[69]

在能够进行严格监管的领域里，例如，在金融系统中，或是在企业构架方面，找到理想的互操作性水平或加强模块化，是有可能实现的。我们可以规定，当机构达到一定规模时就必须进行拆分。然而，事与愿违，在大多数其他类型的技术系统中，各部分之间的互联通常只会继续猛增。当系统规模相对较小时，我们尚能进行模块化处理或分段构建，但是随着技术的发展，这种边界清晰的处理方式将变得越来越不可行。在社会压力和系统传统结构的共同作用下，我们不得不继续强化系统之间的相互联系，以致它们越来越难被分解。因此，尽管我们渴望简单，但现实却背道而驰。

虽然我们也可以尝试构建一些设计得更合理的系统，但这种方法只能在一段时期内行之有效。例如，现在业已出现的“计算机科学和工程实践”[70]，其实质是“工程卫生学”，可以大大减弱系统的复杂性，比如避免计算机程序出现某些类型的变量。如果丰田公司采取了这种做法，那么其汽车系统的整体复杂性就会减弱很多。此外，专业软件通常也会包含一些能够降低代码错误率的设定。这些方法能够将每千行代码的错误率降低至6%，而这显然是一个极低的数字。[71]当人们共同构建、操作和维护复杂的技术系统时，管理团队的一些特殊操作将有助于减少系统问题。但是从长期来看，吸积、交互和边界情况最终会使这些简化工作成为徒劳。

随着时间的推移，系统渐渐变得复杂；而面对这些复杂系统，我们的大脑也渐渐无能为力。无论是互联网，还是大型基础设施，要想从整体上理解它们，已经不可能了。

那么，为什么必然会产生这样的结果呢？在接下来的一章中，我们将讨论人类理解能力的社会极限和生物极限：无论有多么努力，我们的大脑和社会在面对这些复杂系统时的表现都不会太好。

# OVERCOM PLICATED

## Technology at the limits of comprehension

对于大多数人来说，记住一个 7 位数就相当不容易了。当复杂系统的组成部分和连接数量猛增时，即便是专家也会望而生畏。我们个人的知识储备，与理解复杂系统所需要的知识相比，存在着根本性的冲突。

03

## 为什么复杂系统越来越难以理解了

对于大多数人来说，记住一个7位数就相当不容易了。当复杂系统的组成部分和连接数量猛增时，即便是专家也会望而生畏。我们个人的知识储备，与理解复杂系统所需要的知识相比，存在着根本性的冲突。

1985年的一天，一位患者来到一家诊所接受宫颈癌的放射治疗，所用的设备是Therac-25型大型放射治疗仪。[1]操作员按照医生的处方设定好剂量并开始治疗，而后放射治疗仪提示“信息错误”并称“无辐射剂量”。操作员重新尝试了一次，结果还是一样。此后，操作员又尝试了三次，也就是说，他为患者进行了五次治疗，而每一次治疗仪都提示“无辐射剂量”。治疗结束后，患者抱怨髋部有灼烧感，随后被送入医院接受治疗。

几个月后，该患者因癌症去世。后来发现，她遭受了可怕的过量辐

射，即使能够幸存，她的臀部也需要被全部切除。然而，那台机器却说“无辐射剂量”。

这并不是特例，这种型号的放射治疗仪还出过其他故障。20世纪80年代，有6位患者经历了Therac-25型放射治疗仪的错误治疗，在一次治疗过程中遭受了过量的辐射。辐射严重过量对患者身体造成了极大的伤害，甚至导致其中一些患者失去了宝贵的生命。辐射过量事故是这种型号的放射治疗仪问世以来最糟糕的故障。

能不能避免这种故障？或者降到最低？仔细看一下制造商于1983年提供的关于这些仪器的安全分析报告便不难发现，导致故障的原因之一是：参与设计和测试的技术人员只关注了硬件问题，而忽视了软件问题，因为他们认为“软件不会因磨损、疲劳或复制而退化”[2]。虽然这句话大体属实，但他们却完全忽略了一个更重要的事实：软件本身就很复杂，出错的形式也多种多样。这份报告表明，制造商对软件的复杂性缺乏认知，而这种复杂性可能会导致致命的辐射过量。软件缺陷在生活中并不鲜见，但是该制造商的安全分析几乎完全忽略了它们隐含的风险。

显然，因为技术人员对软件的复杂性有所误解，所以Therac-25型放射治疗仪的安全性未能得到保证，换句话说，是那些技术人员酿成了这场致命的灾难。事后看来，他们理应很容易找出问题所在，然而他们低估了系统中各个组成部分的重要性，最终导致了致命的故障。不过，在新的技术系统中，对问题的诊断的确已经变得越来越困难了。无论我们多么努力地构建和设计符合逻辑的技术系统，系统中总会有某些部分是我们无法完全理解的。原因很简单：我们是人。人类的思维方式与复杂系统的运行方式是完全不匹配的。复杂系统的构建方式决定了它们很难被理解，甚至不可能被理解。

在学习编程时，最初要学的技能之一，是用与日常生活中不同的方式进行计算。当然，这并不意味着必须用二进制或十六进制计数，对于

大多数程序员来说，这只是一个好玩、但不必需的技能。我的意思是，程序员总是需要从“0”开始计数，在他们的列表中，第一个对象始终是“0”。之所以要从“0”而不是从“1”开始计数，是因为那就是计算机计数的方式。[3]正如作家斯科特·罗森堡（Scott Rosenberg）所说，计算机计数方式与日常计数方式之间的差别，正是我们需要对计算机代码进行调整的地方之一，同时也是错误和故障的发源地之一。[4]我们必须在计数时调整1个数位，反复增加或减少，以协调人类世界的“编号”与计算机世界的变量枚举之间的差异。如果调整失败，错误就会倍增。

人类世界以“1”为始，而计算机则从“0”开始计数。这个事实意味着：人类的思维方式与大型系统的构建及运行方式之间的裂痕，将会越来越大。我们无法一一跟踪大型系统中的所有过程，以及每一次交互所导致的一系列后果。人类的大脑并不具备解析这种复杂性的能力。在大型系统的构建方式和人类的思维方式之间，我们看到了诸多复杂因素，而这些复杂因素不仅会降低系统的可理解性，还会导致出人意料的后果和问题。

例如，在军事行动中，战士们经常要面对非常复杂的情况。这便要求他们既要具备处理大量信息的能力，又要具备快速的反应能力。但是有些时候，如果情况太复杂、太紧张、太混乱，战士们可能会不堪重负，继而丧失处理紧急事件的能力。据说，当这种情况发生时，战士们会变得“像无头苍蝇一样”（lose the bubble）。正如加拿大政治学家、生态学家托马斯·霍默－狄克逊所描述的那样：“以往不难理解和可以预见的一切，突然之间都变得混沌不堪、令人困惑。”[5]由于对形势的认知急剧下降，战士们已无法处理接二连三到来的刺激信号，更无法采取行动。

这个问题不只会出现在那些身处紧张局势、面对复杂情况的人身上，例如航空交管人员，以及需要在战斗中做出冷静判断的人，其实这

个问题会出现在所有人身上。在面对技术系统时，无论是个人，还是集体，都会遇到这样的问题。人类已经失去了控制权。

在纠缠世界中，我们是因为以下两种相互关联的原因失去控制权的：第一种，无法完全把握庞大而复杂的系统结构和动态，即无法把握系统的不同部分以整体的形式相互作用；第二种，无法获取足够多的必需的、用以理解这些系统运行方式的专业知识。接下来，我们将再一次以语言学为例，力求更清晰地说明，人类的思维方式为什么无法适用于复杂的技术系统。

## 力不从心的大脑

在计算机科学中，“递归”（recursion）这个术语本意为自我引用，它描述了一部分需要引用回自身的计算机代码。这个概念催生了不少程式化幽默。在搜索引擎上查找“递归”这个术语时，它可能会反问你：“你的意思是递归吗？”这对许多人来说无疑是很有趣的。

递归也潜藏在语言结构中。换句话说，语言具有递归能力。事实上，语言在理论上是无限递归的。你可以说“他说狗是棕色的”，也可以说“她认为他说狗是棕色的”。你在第二句话中嵌入了第一句话。更大胆一些，你甚至可以说“我记得她认为他说狗是棕色的”。基于不同的句型结构，在一个句子中嵌入另一个句子的现象，可以出现在句首、句中或句末，而且还可以一遍又一遍地嵌入。

递归，使语言变得无限丰富。举个例子，假设有一个规模相对较小的语言系统，它仅包含1 000个动词、10 000个名词，以及这样一个规则：将单词组成句子的唯一方式是“名词＋动词＋名词”。这种语言看似简单，但应用功能却极其强大。你可以造出10 000×1 000×10 000个句子，也就是1 000亿个句子。就算你以每10秒钟说一句话的速度，时刻不停地说，也需要3万多年才能说完所有句子。[6]更重要的

是，上述事例中的语言系统，其实是一种体量很小的语言系统。如果你面对的是一种单词量更大、句型结构更复杂的语言，想要将所有句子都说一遍，所需要的时间会十分漫长，漫长到需要地质时间来描述。

虽然句子可能会多到不可思议，但其数量终归还是有限的。在上文中，我们讨论了大得惊人的有限性，现在我们需要将讨论转向真正的无限性，不过在此之前，必须再谈一谈递归这个概念。一旦你在一种语言系统中引入了递归，也就是允许将任意数量的子句嵌入到主句中，那么这种语言的丰富程度在理论上已具有了无限性。

然而在现实中，这种说法并不完全正确。说一种语言系统允许任意数量的子句嵌入主句，无疑是愚蠢的。尽管就语法规则而言这是可行的，但是实际上，我们的大脑根本无法解析那么多层的递归。尽管我们希望能拥有无限性和多样化的语言，但我们能处理的递归深度通常不超过三层。

下面这两个例句引自认知心理学家、语言学家史蒂芬·平克（Steven Pinker）[8]的《语言本能》。[7]句子不仅难以理解，而且语法似乎也是错的。

The dog the stick the fire burned beat bit the cat.

一只被火烧过的棍子打过的狗咬了这只猫。

The rapidity that the motion that the wing that the hummingbird has has has is remarkable.

蜂鸟所拥有的翅膀的运动的速度有了显著提高。

这两个句子所嵌入的层级其实都不算多。例如，第一句话的逻辑是一只狗被火烧过的棍子击打，然后咬了一只猫。这句话的语法结构是：对“狗咬猫”中的“狗”进行修饰，同时对棒子进行描述。所以我们说，这句话只有两层嵌套。如果句子的嵌套达到了10层，就不可能有

实际意义了；如果人类连10层嵌套都处理不了，就更不可能处理无限嵌套了。

有一些类型的语言处理任务是人类可以完成的，比如翻译，而且人类可以做得比计算机更好，但是在解析句子方面，计算机则有很大优势。人类的认知会受到大脑工作记忆的限制，但是计算机却可以使用大容量的存储器，将句子的各个部分放在相应的位置上，然后构造出句子树，并使其拥有某种意义。正因如此，计算机才可以轻松地分析那些令人类大脑“崩溃”的句子。例如下面这种奇怪的句子：“This is the cheese that the rat that the cat that the dog chased bit ate.”（这是被狗追赶的猫咬了的老鼠吃的奶酪）。无论是谁，听到这样的话肯定都会头昏脑涨、不明所以，然而计算机却可以轻而易举地做出解析。[8]

有些计算机程序甚至直接内置了语法结构，因而能够创建出层级相当庞杂的句子。不过，它们无法让这种复杂的句子变得合理，只能让它们看上去像是人写的，并且风格与某些作家相似。以“康德发生器”（Kant Generator）[9]为例，它可以造出这样的句子：“由于对现象的知识是先验的，因此，如我所知，对读者来说，就必然性和事物本身而言，应该非常小心地观察，人类理性的规律，可以像我们的判断一样对待。”人类作为生物，比小小的计算机程序复杂得多，但是在拆解问题，并确定其合理性方面，却无法像计算机程序那样信手拈来。

与此类似，还有一种被称为“花园小径句”（garden path sentence）的语法结构。[10]例如“The complex houses married and single soldiers and their families.”这类句子，看上去很有趣，但实际上却充满歧义。这些有趣的句子一开始是单向的，但最后呈现出来的语法结构和意义却完全超出了我们的预料。当句子切换路径时，我们会自动加入既定的含义，但结果往往令人惊讶，并让人一时间摸不着头脑。

当然，除了在面对递归以及其他某些语法技巧时，我们会显得无力之外，人类认知能力的局限性还体现在其他诸多方面。事实证明，许多机器能够轻松完成的任务，人类却无法胜任，这种情况几乎成就了整个家庭手工业。因此，我们无须再讨论“视觉幻象”这类感性挑战了，不如来看看有关心理能力极限的实例。[11]例如，你有没有想过，你能瞬间记住的数字有多长？对于大多数人来说，只能记住7个数字左右，也就是说，你很难记住比电话号码更长的数字，而且这里所说的电话号码是不包含区号的。[12]

又如，你了解自己的计数能力吗？更直接地问，你能数清楚计算机屏幕上有多少个点吗？当然，人类绝对有能力数很多数，但是你一眼就能看清，并能准确地说出来的屏幕上的点又有多少呢？事实证明，这个数字非常小。当你观察一大组点时，大脑会将它们分成多个小组，通常是三四个组。在视觉上，大多数人能瞬间识别的物体只有4个左右。这种对数量的即刻察觉能力被称为感知（subitizing）。这是人类大脑的怪癖之一：只能对少数几个对象进行有效的感知。例如，当我们在阅读冗长而曲折的句子时，通常会遇到障碍，因为我们无法同时识别4个以上的对象。

在与计算机的其他“较量”中，人类的“成绩”也很可悲。将一个信息小片段传输到大脑的长期记忆区大约需要8秒钟，但是在同样的时间，甚至更短的时间里，我们可以将整本《战争与和平》下载到笔记本电脑中。[13]人类在处理多重任务时的效率极差，远远比不上最低配置的计算机。我们的神经元回路比计算机电路要慢100多万倍。[14]另外，据估计，大脑的长期记忆容量甚至还比不上20世纪80年代初出厂的一台老式苹果机（Macintoshes）。

人类的大脑最终能不能克服其局限性呢？通过对人类认知的研究，我们得到的答案并不如意。就像可以加快计算机转速，让计算机超频运行一样，我们偶尔也可以利用药物来“提神醒脑”。但是研究表明，大

脑在“超频”时也会权衡利弊，做出取舍。正如超频运行的计算机可能会发热一样，超极限运转的大脑也会受到伤害。从现有的证据来看，人类的大脑在进化过程中已经微妙地实现了优化，[15]所以随意“超频”可能会致使它出现严重的问题。

只要仔细观察一下那些拥有超强记忆力的“奇人”便可知，这种情况实则是“矫枉过正”。这些人可以记住遇到的每一件事，但是他们并不是超人。事实上，他们会遇到许多困扰，比如，在诸如“面孔识别”之类的简单任务中，他们的表现会非常差。这是因为他们的记忆过于细节化，所以只要人的面孔出现一点点变化，他们就很难进行确认。在阿根廷作家豪尔赫·路易斯·博尔赫斯（Jorge Luis Borges）的短篇小说《博闻强记的富内斯》（Funes the Memorious）[16]中，主人公富内斯拥有完美的记忆能力，同时也因此而背负着无比沉重的负担，因为每一个细节变化都会在他的大脑中形成新的记忆。就像富内斯这个虚构的人物所面临的处境一样，在现实生活中，这种令人难以置信的完美记忆能力会导致其他诸多方面的问题，例如缺乏抽象思维能力。若一个人不得不背负大量的不必要信息，一切就会变得得不偿失。

在这个世界上，的确存在拥有超常认知能力的人。例如，有的人拥有超大记忆容量，有的人能够心算超多位数的加减乘除。既然如此，那么在理解复杂系统这件事上，应该也能找到极具洞察力的人。在这一点上，印度数学家斯里尼瓦瑟·拉马努金（Srinivasa Ramanujan）就是最典型的例子。[17]拉马努金是一位自学成才的数学天才，可惜英年早逝，如同一颗彗星般匆匆划过了20世纪早期的天际。他不是一位普通的数学家。普通的数学家只能依靠反复试错和偶尔闪现的灵感来解决问题，但是在拉马努金那里，方程式会像瀑布一样直接从大脑中奔涌而出，而且除了少数方程式不完全正确以外，绝大多数方程式都复杂和准确得令人震惊。

苹果公司创始人之一斯蒂夫·沃兹尼亚克（Steve Wozniak）堪称

技术研发领域的拉马努金。他不仅完成了对第一台苹果电脑的编程，还负责了第二代苹果电脑的开发。正如程序员兼小说家维克拉姆·钱德拉（Vikram Chandra）所说：“这台计算机的每一个组成部分、每一个字节都是沃兹尼亚克一手打造的，而且人们尚未发现任何错误……沃兹尼亚克一人包办了所有的硬件和软件，还在计算机代码中创建了一种编程语言。他本人就是硬核。”[18]言下之意，沃兹尼亚克对技术的理解无人能及。

事实上，在现实生活中，我们也能观察到大脑的认知极限以及这种极限的扩展方式。伦敦的出租车驾驶员对道路知识的获取和运用，便是一个很好的例子。“知识”是一个奇妙的术语，这里所说的知识是指伦敦地区道路系统的所有细节：25 000条道路及其相互之间的连接点，以及公共汽车站、地标性建筑、雕像、餐馆、酒店……[19]为了将乘客准确高效地送达目的地，出租车驾驶员必须掌握全部细节。积累这方面的知识，成为一名出色的出租车驾驶员，可能需要长达数年的强化记忆和积极的实践。最终，这些出租车驾驶员的大脑出现了明显的变化：对空间记忆至关重要的海马后部脑区增大了。

然而，尽管存在这样一些“异常点”，尽管这些特例给我们留下了非常深刻的印象，但是人类大脑的认知能力终归还是有限度的。[20]拉马努金也会犯错，沃兹尼亚克肯定也有无法突破的认知极限，譬如递归问题。而伦敦的那些出租车驾驶员，显然也不可能把全世界的道路信息都存在脑子里。

如前所述，我们的记忆能力以及在记忆中进行检索的能力，都是有限的，能瞬间识别的事物数量也少之又少。除此之外，我们还很难理解系统内部各种互联的具体含义。具体来说，在面对非线性变化时，我们会不知所措。当某个事物以线性方式发生变化，即一个较小的变化产生一个较小的差异、一个较大的变化产生一个较大的差异时，我们所需要做的主要是以线性形式进行外推。在这样做的时候，大脑几乎不会遇到

什么困难，因为线性系统的输入与输出是成正比的。但是，在非线性变化中，一个小变化在流经一个大型互联系统时，会导致系统以不成比例的方式发生变化，以致大脑无法很好地做出应对。

在非线性系统中，行为会受到反馈和输入放大率调制的影响（或者也可能相反：一个很大的变化只能带来一个极小的影响），这令大脑很难将输入与输出关联起来。在这种情况下，我们无法继续以线性形式进行外推。所有变量之间的互动形成了错综复杂的不规则曲线，让我们的大脑一筹莫展。正是因为大脑本身具有这样的缺陷，所以我们很难把握复杂系统，包括我们自己所构建的复杂系统。

## 认知的极限

哲学是一门泛科学，无论是政治还是伦理，抑或是科技，都属于哲学研究的范畴。在有关科技的哲学研究中，人们对“软件的哲学含义”这个命题越来越感兴趣：我们应该怎样理解软件这种算法作品？为此，美国堪萨斯大学的哲学家约翰·西蒙斯（John Symons）和杰克·霍纳（Jack Horner）研究了这样一个问题：为何我们对软件这种技术系统的构建，几乎立即就会产生无法理解之感。[21]

这种情况之所以会出现，最直接的原因在于：软件中存在大量的分叉点。所谓分叉点指的是：如果条件A为真，那么就做这一件事情；如果条件B为真，那么就做那一件事情。对程序员来说，这叫作if—then语句。例如，有的计算机程序可能会规定，如果某个数是奇数，那么就加上10；如果是偶数，则加上5。

正如西蒙斯和霍纳所指出的那样，一旦某个计算机程序包含了多个分叉点，那么该程序运行路径的数量会倍增。据保守计算，一个只有1 000行代码的程序（即使是对于一个非常简单的程序来说，1 000行也是一个很少的数量；实际上绝大多数程序的代码行数都不会小于这个数

字），当分叉点以正常频率出现时，也会有$10^{30}$个潜在路径可供遍历。$10^{30}$是一个天文数字，超过了1亿亿亿，因此要检查所有可能的路径，理解每个路径的含义，并确定其准确性，不仅是不可行的，也是不可能的。可见，这个系统的确是很难被理解的，即使从宇宙诞生之初便开始研究，也无法掌握它的所有细节。换句话说，绝大多数计算机程序都是永远不可能被任何人完全理解的。这个结论适用于笔记本电脑中的应用程序、厨房设备中的计算机代码，以及用来调度全球飞机的航空软件……

当然，在某种程度上，计算机程序也具有一定的可理解性，而且不需要我们手动遍历所有潜在路径。这正是抽象的主要特点之一（请参见本书第1章的讨论）。在抽象的基础上，对软件进行各种严格的测试和纠错，并遵循软件“卫生准则”，比如，禁止使用GOTO语句或某种类型的变量，就可以有效缓解理解无力的状况。但是我们仍然无法确定，我们是否真的了解所有的影响和结果。无论是使用某个科学模型的科学家，还是操作大型机器的技术人员，或是驾驶最先进汽车的驾驶员，都不得不适应这种“不完全理解”的状态。这是生活在纠缠世界中必须付出的代价。

通过大数据分析，我们可以发现庞大和互联所带来的影响。大数据分析就是用特定算法对海量数据进行处理，以增强预测能力，但有时会以牺牲可理解性为代价。[22]例如，谷歌公司已经开始借助算法来提高效率了，[23]其中就包括通过将诸如冷凝器数量、水泵数量以及室外风速等各种数据信息输入特定的计算机模型，来提高其数据中心的能效。在谷歌公司的博客上有这样一段话：“在像数据中心这样的动态环境中，人类很难观察到所有变量……的相互作用，而计算机则特别擅长发掘隐藏在数据中的潜在故事。为此，数据中心的工程师们将日常运营过程中收集到的信息输入了模型，让模型来帮助他们理解作为凡人可能无法注意到的复杂交互。”[24]

要搞清楚这种大规模技术模型的运算细节是非常困难的。而且，正如《新科学家》杂志（New Scientist）首席技术编辑道格拉斯·希文（Douglas Heaven）所说，即使我们能够做到这一点，也不一定有意义。[25]这种软件系统所给出的选项或答案，并不是按照人类的思维方式得出的，而且常常有违于某些稳定的一般规则或思想。这些决策所基于的路径并非简单的逻辑推理，而是一系列异常复杂的运算。我们置入大量的信息和数据，让庞大的软件系统“生产”出一些东西来。然后，可以得到了一个答案，它可能确实有用，但同时我们也失去了对解答过程的洞察力。从法律体系到硬件设施，在一个又一个的领域中，我们始终在通过与计算机“合作”来驾驭复杂无比的技术。然而，在这个过程中，我们又对自己所依赖的这些系统的运行方式感到困惑不已。

不仅如此，这个过程还在一直加速。在进化计算领域，软件已经可以自行“进化”出问题的解决方案了，但我们对最终方案的形式几乎一无所知。那么，当我们需要一个公式来拟合数据的时候，公式又从何而来呢？我们需要借鉴生物学中的进化思想。先在计算机程序中创建一个由潜在的解决方案组成的“种群”，然后让它们自行进化，也就是让它们重新组合、变异和复制，直到最适合的解决方案脱颖而出。进化算法可以很好地实现这一过程，即使你无法理解它给出的最终解决方案，也没什么关系。

这方面的研究多年以前就已出现，其中一项研究是利用进化算法，设计特定类型的计算机电路。[26]具体来说，先选定特定任务所需的电路类型，然后尝试在硬件中进化出一个解决方案，也就是让一组备选电路混合在一起，形成一个达尔文式的“进化浓汤”。经过多重进化之后，计算机程序就会发现一个最完美的电路设计。这种电路设计有一个奇异的特征：有一部分设计是与主电路断开的；而这对整个电路的功能来说至关重要。这个由进化而来电路设计利用了一些奇怪的物理知识和电磁现象，而这些知识和现象是从未有人想到过和利用过的。

此外，人们还利用一个进化来的公式解决了另一个难题。这个难题一度被认定为是不可能完成的任务。凯文·凯利（Kevin Kelly）在其著作《失控》（Out of Control）中是这样描述这个公式的：“它不仅丑陋，而且杂乱，甚至不可理喻。即使是对数学家或计算机程序员来说，这个进化而来的公式也是不折不扣的‘荆棘路上的柏油娃娃’。”[27]最终，研究者们还是参透了进化生成的代码，但是发现它对问题的解决方式“绝非人类所为”。这种进化生成了新颖的技术系统，但是我们很难理解这种系统，因为我们决然想象不出同样的东西。这种系统与我们擅长构建的那些系统有着本质区别。

在物流领域也出现了一些强大的算法。[28]因为进化生成的运货线路看上去不合逻辑，所以驾驶员们对这些违背直觉的线路很不满意，但最终又不得不臣服于它们的效率。事实早就证明，在国际象棋比赛中，计算机比人类强大得多：计算机能够“想到”人类棋手不可能想到的走法，并取得最终胜利。那些极具计算机特色的走法被称为“计算机走法”，人类棋手几乎从来不会使用。它们看上去“丑陋”，但效果却很好。正如经济学家泰勒·考恩（Tyler Cowen）在其《平均时代的终结》（Average Is Over）一书中所提到的那样，这种走法乍看上去似乎是错的，但其实非常有效。[29]一位参与IBM公司深蓝（Deep Blue）计算机设计的工程师也曾说过，“深蓝”在与加里·卡斯帕罗夫（Garry Kasparov）进行国际象棋比赛时曾用了一着超级怪异的走法，以致“评论员们有的赞不绝口，有的大加抨击”。[30]但是人们后来发现，“深蓝”这一着既怪异又极其出色的走法，其实是程序中的一个缺陷所致。无论如何，这样的人机对弈揭示了一个事实：在最高级别的对弈中，对人类来说，国际象棋还是太复杂了，因为交互的部分太多，即使是像卡斯帕罗夫这样的国际象棋特级大师也无法完全理解。许多时候，我们甚至无从分辨哪个决定是错的，哪个决定是对的。

那么，法律系统的情况又如何呢？在法律系统中，我们也看到了同样的问题，而且这些问题正愈演愈烈。在上一章中，我们曾经引用过马

克·弗勒德和奥利弗·古迪纳夫的观点。除那之外，他们还指出，“但凡对法律进行解释，那么就算解释得再明确、再精准，人们对法律文件的理解过程也会变得缓慢而曲折，[31]因为我们需要不断地核查、重新定义、交叉引用、处理例外……，如此一来也就嵌入了复杂性。但是，律师的智慧——作为传统的契约解释的计算机制，不但非常昂贵，而且会受到认知水平的限制。”因此，当我们发现如下现象时不应感到惊讶：假设我们将同一份家庭收入数据分别提交给45名税务人员，他们会反馈45种不同的纳税方案。[32]

除了系统组成部分及其交互的复杂性超出了我们的处理能力，并会导致我们无所适从之外，摆在我们面前的还有另一个不利因素，那就是我们可以保存在头脑中的知识总量也是有限的。这不仅涉及原始数据，还涉及专业技术知识。技术系统所涉及的知识面变得越来越广泛，而且还跨越了不同的领域，要知道，即使是专家也不可能了解所有这些知识。

为了了解认知的局限性，我们需要先了解一下人类追求专业化的历程。然后我们就会明白，当今这个“不可理解”的新时代，其实由来已久。实际上，人类社会长期以来一直保持着这种进程，只不过现在才抵达终点。

## 最后一个无所不知的人

在我的书架上有三本书[33]的书名或副书名都用到了“最后一个无所不知的人”（The Last Man Who Knew Everything）这个短语。第一本书是讲述阿塔纳斯·珂雪（Athanasius Kircher）的生平和学术贡献的文集。珂雪是一位生活在17世纪德国的耶稣会牧师。今天的人们普遍把珂雪视为一位怪杰，但是也有一些人认为他是一个骗子。珂雪的著作可谓包罗万象，从天文学到埃及象形文字，不一而足，甚至还包括了以猫为题材的音乐剧。第二本书和托马斯·扬（Thomas

Young）有关，扬出生于1773年，研究的领域涉及物理学、医学和语言学，等等。第三本书则是约瑟夫·莱迪（Joseph Leidy）的传记，他生于1823年，住在美国费城，是一位古生物学家和博物学家。

那么，最后一个无所不知的人到底是谁呢？我不知道。事实上，应该没有人能洞晓人类文明所生产的一切。在过去的几个世纪里，知识体量呈现出爆炸式增长，即使只是想要了解所有新知识也是不可能的。同时，在我们眼中，宇宙也变得更加复杂难懂了。不过，在“最后一个无所不知的人”所生活的时代，甚至在更早之前，确实存在不少努力让自己变成无所不知的人。这种努力往往和一种被称为“珍奇柜”（cabinet of curiosities）或“珍奇博物馆”的事物密不可分。

“珍奇博物馆”一词，德语为“wunderkammer”，指的是一种特殊的房子，其主人为了掌握世界上的“全部知识”而四处搜集珍奇事物，最后把房子塞得满满当当。珍奇博物馆可谓是怪异物品的庞杂集合，动物标本、草药、绘画，应有尽有。通常，它们的主人是富有的欧洲贵族。珍奇博物馆既是社会地位的象征，也是了解宇宙及其间奇迹的窗口。那些收藏家对人造事物，例如乐器和武器；以及天然物品，例如化石和矿物，兼收并蓄。他们的橱柜里装着一整个世界，而世界的多样性通过这些摆在眼前的藏品得以集中呈现。正如作家菲利普·鲍尔在其《好奇心》（Curiosity）一书中所描述的那样：“理想的收藏必定是全面的。这种全面性不仅是因为它包含了世界上每一种物体和物质的样本，或者至少是在向穷尽所有事物这个目标靠拢，还因为它是一个完整的世界缩影，也就是说，它以微缩形式表现了整个世界。”[34]

在珍奇博物馆中，你只要看上一眼，就可以窥见整个宇宙及其间的一切现象。但是人们很快发现，有些珍奇博物馆似乎只是大杂烩般的存在；更加重要的是它们永远不够大。菲利普·鲍尔还在书中引用了法国作家帕特里克·莫里斯（Patrick Mauries）的观点，在美洲大发现之后，人们意识到，任何一个博物馆都不再可能包罗世界万象了，譬如不

可能包含所有的物种。[35]自那之后，人们不得不承认，这个世界实在太多样、太复杂了。于是，人们开始做起选择题：哪些东西应该收藏进博物馆？哪些东西可以忽略？

此后，珍奇博物馆仍然存在了相当长的时间。几十年前，我参观过尼亚加拉大瀑布博物馆。它是当时世界上仅存的几家珍奇博物馆之一，其所有者是我一位朋友的父亲。当然，它现在已经闭门谢客了。在“大自然的怪物”展厅中，摆满了千奇百怪的动物标本，有五条腿的奶牛，还有两个头的羊。凝视着那些突变体，以及镶嵌在墙上的昆虫标本和埃及木乃伊，我深刻地体会到了“世界之大，无奇不有”这句话的含义。

但是，这些珍奇博物馆所陈列的事物远非世界之全部，想要包罗万象是不可能的。每一家珍奇博物馆的存在都意味着某个特定的选择过程已经结束。而且，在这种选择中，我们可以看到专业化分工的深化痕迹。随着知识的范畴逐渐超越了地域、文化和心灵的界限，想要掌控好周围的系统，我们就必须求助于专业化。换句话说，我们必须更好地理解细分领域，比如先进武器，或者某个学科的子领域。当然，这种转变绝非一日之功。

在几个世纪以前的很长一段时间里，有许多人仍在试图理解周遭世界，而这种努力不再只是简单的收集过程。事实上，他们都博学多识，例如生活在17世纪后半叶至18世纪前半叶的哲学家、科学家和数学家戈特弗里德·莱布尼茨。根据史学家丹尼尔·布尔斯廷（Daniel Boorstin）的说法：“在26岁之前，莱布尼茨为神圣罗马帝国制订了一项法律改革计划，还设计了一台计算器，并试图游说路易十四不要攻击莱茵兰（Rhineland），而去修建苏伊士运河。”[36]用腓特烈大帝的话来说，莱布尼茨一个人就是“一所完整的大学”。[37]与此类似，牛顿通过万有引力定律将物体的下落轨迹、火星的运行轨道等所有现象都统一了起来。

同一时期，英格兰最古老的学院格雷沙姆学院（Gresham

College）则致力于举办各种公开的主题讲座，并在天文、几何和音乐等学科领域均拥有小型的教师团队。但事实上，格雷沙姆学院的各种头衔并没有太大的实际意义，一些教授会根据可以得到的房子的质量，而非自己的专业特长来选择头衔。[38]在那个时期，专业化并不被人看重。对此，数学家伊萨克·巴罗（Issac Barrow）曾说：“一个不博学的人，不可能是一个好学者。”[39]

然而，时至今日，知识的积累和扩展已经远远超出了任何人所能掌握限度。想要在知识最前沿构建关于世界以及新技术系统的新模型，我们就必须尽可能地做到“对越来越小的领域知道得越来越多”，也就是要在某个特定领域内进行深入的专业化研究。[40]对此，美国西北大学的本杰明·琼斯（Benjamin Jones）提出了“知识负担”理论[41]：想要在前沿知识领域取得进展，你就必须先了解相关领域以往积累起来的全部知识。由于人类的集体知识一直在不断增长，所以“知识负担”只会越来越重。在这种情况下，为了能做出新的贡献，我们不得不进行更多的学习。在另一篇与他人合作的文章中，琼斯还指出，1639年，约翰·哈佛（John Harvard）在临终前将财产和藏书遗赠给哈佛大学的前身，这也是哈佛大学名称的由来。[42]当时，他的所有藏书总共只有320本。而在今天，仅仅是美国国会图书馆就保存了超过3 600万册藏书，以及其他印刷资料。[43]“知识负担”从来没有像今天这样沉重过！当我们试图构建或理解复杂的系统时，不仅需要了解更多，而且还需要掌握日益专业化的知识。

对于这种变化，生物学家爱德华·威尔逊（E. O. Wilson）[9]是这样描述的：

> 1797年，在托巴斯·杰斐逊（Thomes Jefferson）担任美国哲学学会会长期间，所有的美国职业科学家和人文学者都可以舒适地坐在哲学大厅的同一个讲堂里。他们中的大多数人都有能力参与讨论知识世界中的任何一个问题。当时的知识世界还很小，尚能作为

一个整体来把握。而到了今天，他们的继任者——45万名拥有科学和工程学博士学位的专家，如果都来到费城，那这个城市一定会因过度拥挤而瘫痪。一般来说，专业学者只需要在自己所研究的领域内，就专业知识和研究议程与同行进行切磋；而一名成功的学者则需要在膜生物物理学（membrane biophysics）[(10)]、浪漫主义诗歌、美国早期历史，以及其他类似的正式研究领域内开启自己的职业生涯。[44]

除了知识在扩展和分化，学者的数量在增加之外，各研究领域的专业化程度也在大幅提高。

我们身陷两难境地。为了更多更好地了解这个世界上的复杂系统，比如人体，我们需要将传统医学分解为众多专业化的医学学科。与此同时，我们正在构建的系统，或者说使世界运行起来的技术，却日趋庞大和复杂。于是，我们又不得不将许多不同的专业领域“缝合”到一起。举例来说，金融系统的构建需要物理学家参与；计算机系统的开发也需要经济学家参与。又例如，无人驾驶汽车的设计有赖于软件、激光、汽车工程、数字测绘等领域的专家的通力合作。[45]

换句话说，专业化帮助我们不断取得进步，同时我们也更加依赖于跨领域“汲取营养”的系统。这就要求我们必须对所有相关领域了如指掌。然而在今天，任何人都不可能拥有全部知识。这也就意味着，无论对谁来说，这些系统在整体上都是不可理解的。

解决方案之一是推进多学科和跨学科的团队合作：将不同领域的专家组织到一起，这样就有可能在前沿地带取得突破，进而构建出特别强大的复杂系统。在软件开发领域，尽管有一些技术系统是由一个人，或一个小团队创建的，但在更多的时候，创建工作需要大规模团队的长期合作。不仅如此，在这个过程中，会不断有人加入和离开。如果将团队工作可视化，[46]也就是将关键软件的开发过程用信息图表的形式呈现出

来，我们就会发现，这个过程看起来就像是一个由文字、会议和分叉组成的错综复杂的流动集束。在分叉点上，不同的个体来来去去：加入进来，参与软件开发，共同处理不同的文件，然后离开。因此，作为这种过程的产物，软件不仅非常复杂，而且往往极其庞大，以致几乎没有人能够完全理解。我们会得到这样的结果并不奇怪，因为完全了解某个特征的人可能早就离开团队了。

专业化无疑是一个成功的进程，它给我们带来了大量的、令人印象深刻的技术，但同时也将我们带入了纠缠世界。在纠缠时代里，我们不得不依赖于生而为人终无法拥有的、复杂的技术系统知识。事实上，没有人能拥有这种知识。人们千方百计地想要走出这种困境，譬如，有人说，是时候召回博学者和多面手了，应该让他们在当今时代重获新生。我们将在后文中再度审视这种可能性，在这里，我们必须先认识到：在个体所能处理的知识体量，与其需要了解的、与生活息息相关的系统知识体量之间，存在着一条无法逾越的鸿沟。

不幸的是，我们常常忽视这种不匹配；等醒悟过来，为时已晚。我们构建了大量复杂的技术系统，并确信构建的基础是符合逻辑的，直到它们迫使我们面对出人意料的结果——错误和故障，而这些结果会导致诸如全球金融市场之类的重要系统陷入混乱，甚至崩溃。有些系统会做出一些离奇的行为，这些行为甚至连设计者本人都不曾预料到，这样的系统被称为“技术狼人”。在计算机科学家小弗雷德里克·布鲁克斯眼中，大型软件项目自带难以控制的“狼人”倾向：“在我们的民间传说中，有很多令人噩梦连连的怪物，其中最恐怖的就是狼人，因为他们会出其不意地从熟人变身为恐怖的怪物。”[47]

当今时代，“狼人”就是我们自己所构建的系统突然出现的意外行为。它是所有使系统变得复杂难解的邪恶力量的集中体现。在接下来的一章中，我们就来讨论一下相关的问题。


bug并不都是能够找到确切起因的那些错误，在以连接和交互为特征的复杂系统中，经常会出现一些令人费解的bug。尽管我们对这些bug没什么好感，但它们却是这个纠缠时代中无法回避的存在。

在20世纪80年代，游戏《小蜜蜂》（Galaga）非常受欢迎。这是一个经典的射击游戏，游戏者可以通过操控自己的飞船来射杀所有“敌人”。作为“古老”的游戏之一，它不仅画面单一，而且配音笨拙，然而这一切丝毫没有阻碍它的流行。不仅如此，它还有一个有趣的bug。如果你在进入游戏后立刻消灭了绝大多数的“敌人”，并在接下来的大约15分钟内避开了剩下的“敌人”，那么，这些“敌人”就再也无法对你开枪了。这是一件奇怪的事情，尽管游戏者常常利用它来取得高分。[1]

为什么会出现这样的情况呢？许多人认为，这是因为在某些特殊时

刻，控制“射击”的代码出了故障，“忘记”了刷新。不过也有人说，这是开发者有意设置的一个功能，目的是便于开发者进入街机并获得高分。虽然后面这种猜测不无道理，但是我认为可能性不大，因为在其他游戏中，这种所谓的隐藏功能和作弊入口会“表现”得“干净”得多。这可能就是一个bug。

这个bug，以及其他无数类似的bug表明，我们并不完全了解自己所构建的系统。要想了解清楚在《小蜜蜂》这款游戏内部究竟发生了什么，为什么会出现问题，我们需要付出巨大的努力。尽管这款游戏的图形和规则看上去都很简单，但这并不代表它的复杂程度只是仅此而已；只有在它出现故障时，游戏者才能洞察到它的“真正实力”。在这里，bug不仅会指出需要修复的问题，还会告诉我们，我们正处于纠缠时代。[2]

早在1950年，艾伦·图灵（Alan Turing）就已指出，机器能够并且肯定会通过它们的行为，给人类带来重重“惊喜”。[3]现在看来，这种“惊喜”发生的频率将会不断加快。随着人类所构建的系统越来越复杂，系统的实际行为和预设行为之间的分歧也会越来越大。这种超出预期的行为，正是前文所述问题的具体表现。通过上一章的讨论，我们已经清楚，人类大脑与生俱来的若干局限性进一步推动了系统向复杂难解的方向发展。这就意味着，我们所有人，包括终端用户和业内专家，在看到火箭升空后很快自爆，或是精心制定的不同法律条文产生冲突时，都会倍感惊讶。错误和故障是系统复杂性的不良产物，是超预期的、不被接受的东西。

我们在上一章中引述过哲学家约翰·西蒙斯和杰克·霍纳的观点。他们对软件的开发过程进行了深入地研究，并重点关注了软件系统中那些通常不会被检查，以及不会被详细检查的方面。例如，在进行计算时，数据如何存储，怎样四舍五入，等等。这些最容易被掩饰、被忽略的方面，往往最容易滋生大量问题。在一个被广泛使用的引力模拟器

中，人们发现，许多错误都与一种特定的数字处理方式有关。具体来说，在该程序中，一段只有3万行的代码，出现了大约1万个同类错误，而将这些错误修复之后，模拟结果便大为不同。[4]

一而再再而三，我们发现，正是日益增强的复杂性导致了技术系统故障频出。当一种技术变得异常复杂之后，“狼人”就会出其不意地出现，并带来未知的影响。[5]在这些bug中，有一些错误是怪诞的，不过无伤大雅，譬如《小蜜蜂》游戏中的那个小bug；但是，也有很多错误是极具破坏性，甚至致命性的，例如“心脏出血”（Heartbleed）这种漏洞。这个错误主要出现在加密软件中，两年多以来，它一直威胁着全球2/3以上在线网站的安全，包括了Facebook和谷歌。

当然，说到bug，就不得不提到微软公司的Windows。1996年，Bug Detective出版了《Windows 95 Bug大全》（The Windows 95 Bug Collection）。这是一本专门讨论Windows 95的系统bug以及潜在解决方案的书。[6]这本书针对各种bug提出了一些解决方案，这些方案要比其他方案更容易管理。例如，Windows的某个版本在启动时会显示错误提示，而解决方案是：直接忽略提示信息。还有些bug的解决方案则需要更加激烈的干预手段。例如，如果你的计算机安装了某种专用控制卡，那么某些程序就可能无法在Windows 95系统中正常运行。那么，他们建议的解决方案是什么呢？答案是：“选择一台速度较慢的电脑，或者使用不同的柏努利控制卡（Bernoulli controller card）。”显然，这个问题并没有得到根本解决，或许，这个问题本就是无解的。

当系统庞大到一定程度时，上述情况就会发生。系统会以意想不到的方式与用户、其他系统，以及自身进行交互。事实上，在软件规模日益增长的同时，错误率也在大幅增加。不过，你不能就此认为，软件规模翻倍，每千行代码中的错误数量也只是会翻倍而已。[7]事实绝非如此。据估计，和拥有5 000行代码的程序相比，拥有1万行代码的程序

在错误数量上是前者的4倍。

这种不可预测性和脆弱性，实际上是我们所构建的复杂系统的标志，虽然复杂系统对预料之内的冲击通常拥有令人难以置信的稳定性。这里所说的“预料之内”指的是系统拥有针对某种特定冲击的设定，但是在面对预料之外的冲击时，复杂性就会变成一种负担。

为了能更好地理解这种特殊情况，人们开发出了一个数学模型，即高度最优化容限（highly optimized tolerance）模型[11]。虽然系统在经过优化后可以适应各种各样的情况，但是任何“新异事物”都有可能让它们出现灾难性的故障，甚至崩溃。以波音777为例，[8]这种大型飞机是一台极其庞大的机器，包含了150 000多个子系统模块，所有模块指向的目标都是：确保正常飞行，并应对各种情况。但是很显然，它无法应对所有的意外情况。据业内专家称：“波音777在应对大规模的气流干扰、载重和燃料的变化、边界层的湍流流动，以及材料的老化与不均匀等情况时，具有很强的稳定性，但这种稳定性可能会因少数超大规模的集成芯片的细微变化，或者某些软件故障而失效，从而导致灾难。”[9]换句话说，随着系统变得越来越复杂，再细微的刺激都有可能引发灾难。其实我们根本不知道未来可能会发生什么。

事实上，这些意想不到的后果与边界情况和例外情况有关。世界很大很复杂，所以需要一个更大更复杂的系统来管理它。很多情况虽然都具有偶发性，非常罕见，但是却极有可能导致技术故障，因为总体数量实在太多，而且无法被一一测试。还是以丰田汽车为例，我们不可能对它的软件系统进行全面且彻底的测试。正如计算机科学家菲利普·库普曼所说：“常规的车辆检测根本不可能找出所有不寻常的故障。”[10]这就好比，一个人就算穷尽一生也不可能遍历所有可能会发生的事故。

在超越大脑极限的复杂世界粉墨登场之后，噩梦随之而来。当然，这场噩梦并不是指具有自我意识的天网（skynet）已向人类宣战，而是说系统已变得越来越复杂，越来越混乱，以致各种故障接踵而来，不

管人们能否预料到。复杂性注定会带来意想不到的后果，而我们却只能在问题出现时才意识到。

## 并不是所有bug都能被消除

哥伦比亚大学城市发展学教授凯特·阿舍尔（Kate Ascher）出版过一系列有关城市建设、交通网络和大型建筑的著作，并在书中讨论了这些系统的复杂性。[11]这些著作附有很多信息丰富的图表，细节描述也很生动有趣，但是读者读起来仍会感到些许压抑。所有这些系统都已历经了几十年，甚至几个世纪的吸积。随着时间的推移，新的部件一层层地被叠加上去，从交通网络到新能源网络，再到物流网络，皆是如此。例如，为住宅和公共场所供水和排水就是一件非常复杂的事情。为了能有一个直观的印象，让我们来看一下纽约市排水系统的巨大规模：仅在市区范围内，地下管道的总长度就超过了9 700千米，而这还只是这个精心设计的排水系统的一小部分；事实上，这个排水系统的日排放量超过37亿升。

然而，通常只有在出了问题之后，我们才会意识到系统的复杂性。2010年春，因为一场事故，波士顿大都会区的居民们接受了关于如何管理和分配水资源的“速成培训”。在那年5月的第一天，马萨诸塞州韦斯顿市的一条主供水管破裂，[12]而该水管的水是从阔宾水库（Quabbin Reservoir）输送过来的。在接下来的几天里，包括我当时居住的布鲁克林区在内的许多社区的居民都收到了通知：必须把水烧开后才能饮用，因为“现在是用备用水源在供水”。除了对面的剑桥市，水库周围的城镇无一幸免，这是因为剑桥市拥有自己的独立水源。市政工程人员当然知道排水系统的复杂性，但是对于大多数城市居民而言，只有在系统发生故障之后，才会认识到这个事实。

安德鲁·布卢姆（Andrew Blum）在其著作《管道》（Tubes）中对互联网的物理基础设施进行了探讨。[13]他以亲身经历作引：有一

天，他家里断网了，原因是后院的网线被一只松鼠啃断了，于是他开始琢磨互联网的物理连接特性，也就是一种纵横交错的有形网络。

在开源软件开发行业，“林纳斯定律”（Linus's Law）被许多人奉为真理。这个定律是以Linux系统的创始人林纳斯·托瓦兹（Linus Torvalds）的名字命名的，意思是“只要给予足够的关注，所有的bug都不是问题”。换句话说，如果能有足够多的人去检验某种技术，那么任何故障，无论它有多么复杂，看上去有多么难以处理，都是可以修复的，因为总会有人能找到解决办法。

但是，随着系统变得越来越复杂，这个“真理”似乎不再成立了。并不是所有的bug都可以消除：当我们面对复杂的充满交互的系统时，发现并消除每一个bug的可能性微乎其微，并且每一次修复都会引发新的问题。[14]这听起来非常令人沮丧，但在某种程度上来说确实是这样的。幸运的是，我们至少还有一丝摆脱部分困境的希望。

“技术狼人”不仅是人类跨入新时代的标志，同时也为人类指出了管理复杂系统的新方向。波士顿大都会区的供水危机说明了，自来水不是“自动来的”，它也有真实的源头。不断地检测bug是我们了解这个纠缠时代的性质，并保证自身在此间繁衍生息的有限选择之一。

## 从错误中学习

几年前，谷歌公司的电子邮件服务系统Gmail出现过一次严重的服务中断故障，导致许多用户在大约18分钟内无法登录邮件系统。[15]调查结果表明，问题出在谷歌软件的一个小更新包上。那是一个用来平衡邮件处理流量，以保证整个系统不会有任何一部分过载的软件包。软件包中的错误导致许多运行正常的服务器被认定为不可用。虽然这个错误并没有影响到谷歌的其他服务，但是由于Gmail需要特定的数据中心信息，所以它直接崩溃了。引发这个级联式故障的是一个很小的问题。对

于这种级别的小问题，很少有人会预料到它会导致如此严重的系统故障。由此可见，系统中隐藏的某些互联性只能在发生故障时才会显现。

在调试一项技术，或者在试图根除某个错误时，你会发现，系统的实际运行方式与你所期望的大为不同，无论是在汽车软件、互联网安全程序，还是城市基础设施中。在某些情况下，出现的错误都很简单，很容易被理解和修复；但是在更多的情况下，错误是不易被觉察的，甚至几乎不可能被诊断和修复。重要的是，对这种错误进行分类和编目，恰是我们研究复杂系统某个部分的第一步。这部技术世界的“博物志”至关重要，就像博物学家走进大自然、研究大自然，将物种及其复杂性分类编录一样，我们也需要对技术采用类似的研究方法。

我们将会越来越需要“技术博物学”。在这里，不妨再来看一个有关编程的例子。假设，你心中想着某个介于1至100之间的数字，而我的任务是尽可能快速并准确地猜到它。我会先问你，这个数字是不是超过了50；如果是，那么我会接着问你，它是大于75还是小于75……以此类推，我不断地将剩余数字一分为二，直到猜出你心中所想的那个数字。这种方法被称为“二分检索法”（binary search），其核心是将待搜索的数字分为两组。众所周知，这是一种在大型的有序列表中查找所需内容的高效方法。

在整个软件世界中，二分检索法的不同实现形式随处可见。因此，我在2006年读到谷歌公司所发布的一篇博客文章时深感不安。[16]这篇文章讨论了二分检索法的多种实现形式，并指出了它们共同的失败之处：“号外！号外！请务必认真阅读本文内容：几乎所有的二分检索法……都崩溃了！”

虽然在普通软件中执行二分检索法的代码通常都能正常运行，但是事实证明，在涉及大量数据的情况下，二分检索法的许多实现形式都有可能会失败。那篇博客文章这样写道：“当数组的长度，也就是数组元素的个数达到了$2^{30}$及以上时，或者说多于10亿个元素时，二分检索法

就会出问题。如此海量的数据在流行《编程珠玑》（Programming Pearls）的20世纪80年代，是不可思议的事。《编程珠玑》是一本介绍计算机编程方法的经典著作，它所提出的二分检索法的实现形式也包含了这个错误。但是到了今天，在谷歌公司和其他地方，海量数据都已经很常见了。”

直到今天，在庞大的数据集随处可见的世界里，我们终于有机会发现这个错误，有机会更好地了解由我们一手打造的这个特定系统。通过这个bug，我们可以进一步地了解“技术实际上是如何运行的”，而不只是停留在“我们希望技术如何运行”的层面上。[17]

还有很多其他例子也表明了，虽然技术系统为意想不到的行为提供了便利，但同时也让我们有机会了解到技术实际上是怎样发挥作用的。例如，1982年初，温哥华证券交易所发布了自己的股票指数，其编制方法与标准普尔500指数（S & P 500），以及道琼斯工业平均指数类似。这个股票指数的初始点位定在1 000点，然而在接下来的两年内，点位却持续下跌。到了1983年底，点位已经跌至原来的一半左右。这样的下跌是完全没有道理的，因为20世纪80年代早期的股票市场是一个大牛市。那么，这个指数为什么会持续下跌呢？有关方面展开了调查，最后发现这个指数的计算方法是错误的：它会直接把小数点3位之后的数字全部舍去。例如，如果计算结果为382.452 7，那么被存储下来的数值会是382.452，而不是四舍五入后的382.453。[18]

这个过程每天都要上演数千次，正确的数值无处可寻，这意味着损失了很大一部分价值。1983年11月，这个错误终于得到了纠正：前一周周五，指数收盘于500点左右，但是到周一开盘时，指数已超过1 000点，失去的点位重新回来了。这个计算过程中的深层问题，准确地说是算法中的错误，在这样一个异常现象中被注意到。事后看来，这起事件——当股票市场向好的时候，指数却在下跌，其实并不微妙。

我们还发现，企业和政府在利用复杂的机器学习算法，对海量数据

进行挖掘时，也出现了不少类似的失败案例。在大多数情况下，这些机器学习系统对于公众，甚至对于专业人员来说，都是难以理解的“黑箱”。但是有时候，这些大型系统会表现出一些奇怪的或是令人担忧的“举动”，从而无意间为我们提供了探索系统内部世界的线索。

以微软的人工智能聊天机器人Tay为例。根据设计，这个机器人以19岁女子的身份与用户进行互动。微软将“她”放上了Twitter上，然而还不到一天的时间，“她”就已经“成长”为一名种族主义者。这是因为“她”所使用的算法接受了那些来自互联网的充满暴力的反馈信息。显然，Tay的偏执倾向并不是事先设定好的。通过这次失败，微软的设计师们更清楚地看到，程序是如何与原始的、未经过滤的互联网账号进行交互并结出恶果的，而这种恶果也是他们始料未及的。

显然，要想发现bug，“守株待兔”是远远不够的。许多技术开发人员都会主动搜索bug，并将它们集中放入数据库中，以系统化的方式加以解决。

更重要的是，在软件开发领域中，人们开始努力尝试打破这个怪圈。他们开始测试边界情况，搜集用户实际可能做出的各种怪事，而不再局限于预设固定的软件使用方式。例如，奈飞公司通过这种策略合理地开发出了“混沌猴”（Chaos Monkey）。[19]“混沌猴”的功能非常简单：它会出人意料地使奈飞系统服务中断。这里面的基本思想是：只有在观察到了庞大的奈飞系统如何应对各种故障之后，工程师们才能想出办法来维护系统的稳定，并以此抗御突如其来的各种意外情况。奈飞公司希望，在“混沌猴”完成了其使命后，工程师们所设计的系统运行方式，与实际的系统运行方式能够完全匹配。

从失败中吸取教训，是理解任何复杂系统的重要机制。回顾一下科学史便不难发现，几个世纪以来，博物学家们一直在利用这种方法研究自然界中的复杂系统。

# 像生物学家一样思考

在我年纪尚轻时，对我而言，在所有词汇中，最令我感到困惑的一个词便是“杂项”（miscellaneous）。“杂项”这个词看起来是如此不可思议，我常常惊讶于它何以能够存在。这个词的有趣之处在于，它看上去像是用多个不同的语言单位拼凑而成的。我甚至不清楚它的准确发音是什么，但是它确实非常吸引人。

这个词的魔力不仅仅在于它的拼写和发音。看似混乱的拼写早已透露出它的含义：生活中，总有些地方是杂乱的。“杂项”意味着，即使是混乱无序，也可以构成一个类别或一种组织方式。“杂项”的存在本就是一种肯定：无论有多么不规则、多么杂乱，混乱的“秩序”是有可能被容忍和接受的。

接受“杂项”，即接受一种能容忍“杂乱”的精神，并不是每个人都能轻松做到的。在面对某个复杂情况时，包括我自己在内的许多人，本能反应是想要以某种方式去简化它，去除所有杂乱因素，并找出隐藏在背后的优雅结构。这种方法一旦成功，就会给我们带来极大的满足感。当我们找到导致失败的单一原因时，情况就是如此。但是，如果这种方法不起作用，我们就只能直面一片混沌，到那个时候，很多人都会感到不知所措。

与一般人不同，以研究世界“博物”为业的博物学家们，早就习惯了面对“杂项”。当他们发现正在观察的动植物的生活习性和行为习惯符合某种秩序时，他们会倍感欣慰；当然，即使不符合某种理论上的秩序，观察也会为他们带来其他收获。因为缺乏完整的理论框架，他们或许无法解释所见到的每一种生物，但是他们会记录下每一个细节，并尝试着去理解。有一个有趣的反例：某次，一位年轻人请物理学家恩利克·费米（Enrico Fermi）说出粒子物理学所研究的诸多粒子的名字，费米回答说：“年轻人！如果我能记住所有这些粒子的名字，我就是一

名植物学家，而不是物理学家了。”[20]博物学家是什么样的？比如约翰·詹姆斯·奥杜邦（John James Audubon），他对美国境内的鸟类进行了分类和绘图。与物理学家不同，博物学家认为应该了解每一个物种的细节，就算不知道所有物种之间的相互适应性，至少也要知道它们的名字，这是十分重要的。

在自然界中，也唯有通过研究生物进化中的错误和故障，譬如突变和疾病，我们才有可能了解生命系统的奥秘。例如基因复制过程中出现的错误，从染色体中的大型畸变，到脱氧核糖核酸（DNA）中不正确的代码复制，以及它们所导致的、可见的差异或缺陷，都是我们了解基因功能的突破口。研究果蝇的基因突变，有助于我们深入理解生命体如何从单细胞发育而来，以及基因蓝图如何培育出完整个体。具体来说，生物学家破解控制身体形态的关键基因序列的途径之一，就是观察一种可怕的触角足突变体，即一种在原本应该长触角的地方长出了腿的苍蝇。

对于技术系统，我们也需要采用同样的方法。“数字世界”是技术史学家乔治·戴森（George Dyson）[12]提出的一个术语，这个世界正在超越人类的控制范围。根据戴森收集的数据，在1953年3月，世界上只有53千字节的高速存储空间（RAM）。[21]时至今日，一台个人电脑所拥有的存储空间已是这个数字的10万倍有余。数字世界还在继续变得更丰富、更庞大、更互联化。它的发展速度是很多人都未曾想象到的。不仅如此，数字世界正变得越来越独立于人类。在全球范围内，信息的传播速度比我们能识别得更快，而且一直在以奇妙的、出人意料的方式相互缠绕着、作用着。

虽然无法理解所有技术系统之间的所有交互，但这并不妨碍我们成为“技术博物学家”。我们可以将系统及其各部分的多样性记录下来，并加以分类。即使无法完全理解整个系统，我们也可以通过考察异常情况和分析故障等方式，获得卓越的见识。


复杂的技术系统更接近生物学系统，因此，用生物学思维思考复杂技术是个不错的选择。为了从整体上理解系统，我们也会忽略掉一些细节，这时，物理学思维才是首选。我们真正需要的是经过物理学思维锤炼的生物学思维。

17世纪中叶，一位名叫纳撒内尔·费尔法克斯（Nathanael Fairfax）的英国医生在科学杂志《哲学学报》（Philosophical Transactions）上连续发表了几篇论文。[1]费尔法克斯观察到一些很有趣的现象，并决定通过这些论文把他的发现告诉同时代的科学家们。

其中一篇论文的标题是《人类自然特性潜移默化的实例：无论是在人身上还是在野蛮人身上》（Divers Instances of Peculiarities of Nature, Both in Men and Brutes）。在这篇论文中，费尔法克斯讲述了一个40岁左右、习惯喝热啤酒的男人的故事。有一天，这个男人

在喝了一杯冰镇啤酒后病倒了，并在几天内死去。费尔法克斯就此推测：人的胃很可能只能适应一定范围内的温度。费尔法克斯还写了一个女人的故事：每次听到雷声，她都会觉得恶心。不过，费尔法克斯没有去推测为什么雷声会对这个女人产生这样的影响，只是指出：“这位女士从孩提时代起就一直如此。”[2]

我们现在知道，尽管费尔法克斯没能实现他的目标——从记录下来的这些观察结果和事实中总结出某种理论，但是他的观察本身就极具价值，至少这是“理解”的第一步。

在同一时期，年轻的牛顿正在思考物体如何移动，以及光线如何传播。牛顿于剑桥大学三一学院求学期间，一场瘟疫席卷了整个英国。为了预防疫情，剑桥大学临时闭校。于是，牛顿回到了家乡，在伍尔斯索普（Woolsthorpe）的农村生活了几年。[3]也就是在那个时期，他对微积分、光学和行星运动规律等方面的研究取得了根本性的进展。他研究了数学推理和演算，还做了一些实验，比如为了分析颜色的性质而在自己的眼窝上插入一枚长针，以及观察苹果如何从树上掉下来，等等。与费尔法克斯一样，牛顿对观察所得的材料进行了分类和编目；不同的是，牛顿还总结出了一套支配物理世界的定律，并用数学公式将它们描述了出来。

在某种意义上，我们可以说，费尔法克斯和牛顿在同一时期、同一个国度所进行的不同研究，代表了两种理解宇宙复杂性的方法，而这两种方法还存在着竞争关系。

牛顿试图将观察到的所有不同事物都统一起来，也就是通过一组优雅的解释来简化世界的多变性和多样性。通常，他所利用的只是几个公式或定律而已。关于这一点，我们可以在牛顿发现的万有引力定律中看得非常清楚。这个极简公式反映了从物体坠落、潮涨汐落到行星运行等诸多现象的普遍规律。而在今天，物理学家之所以会孜孜不倦地探寻能够一统天下的万用理论（Theory of Everything），也是出于与牛顿相

同的愿景，希望能够发现可以作为人类已知的、宇宙各方面基础的秩序；并让宇宙的每个组成部分都各归其位，将它们放在适当的位置上。科学家托马斯·亨利·赫胥黎（Thomas Henry Huxley）有一句名言，“科学的巨大悲剧”[4]是“一个丑陋的事实往往会杀死一个美丽的假说”。他的意思是，优雅的理论是科学的目标，当某个事物与优雅的理论相悖，或令理论复杂化时，科学便会遭遇最大的悲剧。

费尔法克斯则放弃了对理论优雅性的追求，转而拥抱了多样性和复杂性。即使世界在一定程度上是混乱的，他也愿意接受，并为“又了解到新的细节”而欢欣鼓舞，哪怕这些细节很难立即被融入某个单一的理论框架中。有些人把他的方法戏称为“蝴蝶收藏家”式的方法，即收集多种多样的“蝴蝶”并加以描述。在这里，我们还能发现现代医生的影子，他们是费尔法克斯的智识后裔，为人体各个层级上的完美功能而啧啧惊叹，例如血液凝固过程中的复杂步骤、酶级联反应（enzyme cascades）的复杂性质，等等；还有那些天文学家们，会为强大的太空望远镜所揭示的诸多星系类型而深深倾倒。

博物学家不可能赞同赫胥黎的抱怨，因为在他们看来，根本不存在所谓的“丑陋的事实”。所有的事实和知识都为我们提供了与这个奇妙世界有关的新信息，向我们展示着世界的复杂性和多样性。当事实不符合我们的心智模式时，完全不必为此而感到沮丧；相反，还应该为这种“意外”而感到由衷的高兴，然后去寻找能够解释这些“意外”的新方法。

物理学家弗里曼·戴森（Freeman Dyson）将牛顿的方法描述为古雅典时期的科学思维方式，并认为这种思维方式“强调思想和理论……试图找到可以将宇宙万物联系在一起的统一理论”。[5]至于关注多样性的方法，戴森则认为，可以将它描述为工业革命时代的科学思维方式，“强调事实和事物，试图探索和拓展人类对自然多样性的认知”。例如，起源于曼彻斯特的英国工业革命。

戴森还进一步指出：这两种方法还有另一个不同之处，即生物学是多样性理论者的“领地”，而物理学则是统一理论者的“主场”。我在这里把这两种方法分别称为生物学思维和物理学思维。在物理学中，人们通过统一和简化去观察各种现象的明显趋势，无论是在爱因斯坦、牛顿，还是麦克斯韦身上，都能看到这一点。众所周知，麦克斯韦给出了能解释电磁原理的公式。简化，甚至极简化，是物理学领域内广受尊崇的方法之一。

生物学家通常更愿意接受多样性，并倾向于陈列大量事实，而不在意这些事实是否能用某个统一理论来解释。[6]事实上，他们只需要有一个合适的小模型就可以了。当然，生物学理论也并非总是如此。例如达尔文的进化论，显然就是生物学中的一股统一力量；许多分子生物学家、以数理生物学为专业方向的应用数学家，以及许多研究其他领域的生物学家都倾心于此。

说到底，上述两种方法都是在探求具有普遍性的、有预测能力的理论。但是，这两种思维方式的推进方向是不同的，这种不同主要反映在它们对抽象化的相对容忍度上，而相对容忍度又取决于所研究系统的特性和复杂性。例如，利用数学公式抽象掉宏观层面上的细节，这种做法在物理学中几乎无处不在，但在生物学中却难觅踪影。

从下面这个古老的科学笑话中，我们可以清楚地看出这种区别。一位奶农为了提高产奶量雇用了两个顾问，一位是生物学家，另一位是物理学家。生物学家在考察了一周后，提交了一份详细的长达300多页的报告，写明了每头奶牛的产奶量具体取决于什么，例如天气情况、奶牛的大小和品种等。而且这位生物学家还向奶农保证，只要严格按照建议执行，奶牛的平均产奶量可增加3%至5%。而物理学家只考察了3个小时就回来了，然后宣称自己已经找到了一个能够适用所有奶牛的高效解决方案，并且可以将产奶量提高50%以上。奶农问：“那么，你说应该怎么做呢？”“好吧，”那个物理学家回答道，“首先，假设你有一头

身体为球形的奶牛……”[7]

抽象化方法当然是有用的，但我们不能做出存在“球形奶牛”这种假设。当你把生物学层面上的细节都抽象掉之后，你不仅会丢失大量信息，而且最终还会对某些重要组成部分，比如边界情况，感到束手无策。

生物学思维和物理学思维是解释世界的两种不同方法，适用于不同的系统，而且通常是互补的。

## 复杂的技术系统需要生物学思维的3个原因

那么，我们应该如何对待复杂的技术系统呢？它们是生物系统还是物理系统？理解技术系统应该采用哪种思维方式？现在我们就来探索生物系统和物理系统的特征，并与我们所了解的技术系统特征进行比较。

复杂的技术系统需要生物学思维有以下3个原因。

第一，生物系统通常比物理系统更复杂。

在物理世界中，系统的组成部分一般是相同的，也就是说，物理系统通常是这样的：由相同的气体分子组成封闭系统，或是由同一种原子组成单质，比如钻石。此外，在整个系统中，各部分之间的相互作用方式往往是统一的，例如卫星围绕行星运行的方式。

但是在生物学中，情况却不是这样的。在生物学中，系统的组成部分不仅类型繁多，而且涉及很多层级。例如，细胞中的蛋白质具有多样性；生物个体内部的组织器官也功能各异。要研究蓝鲸的交配行为，海洋生物学家就不得不考虑一切与之相关的事情，比如它们的脱氧核糖核酸，以及海水的温度等。生物系统中的每个组成部分不但各不相同，而且还很难从整体中被单独拆解出来。例如，要想观察一只变形虫的细胞核，并尝试分析它的特征和功能，就需要让细胞核留在生命体内，只有

这样才能了解细胞核如何适应变形虫的生活，如何提供涉及细胞诸多功能的核心遗传信息。现在，我们的技术系统正变得越来越复杂，显然，它们更像生物系统而不是物理系统。

第二，生物系统有别于物理系统的一个重要因素是，生物系统是有历史的。

生物会随着时间的推移而进化。尽管物理学所研究的对象也不是无中生有的，天体物理学家甚至还经常讨论恒星的进化进程，但是相比之下，生物系统更容易受到进化的影响。实际上，这正是生物系统的根本特性之一。生物系统之所以会呈现出如此复杂的结构，正是因为系统中存在极为复杂的历史路径，而且，在漫长的进化进程中，历史路径还会受到众多因素的影响。因为生命体的形式很复杂，所以任何微小的变化都有可能会带来意想不到的后果。随着时间的推移而发生的变化，一直在做着修补工作：以零敲碎打的方式修整着系统，使之适应新的环境。

举个例子，人类细胞中有许多至关重要的脱氧核糖核酸序列，譬如那些为遗传密码的翻译工作提供“指导”的序列，还有那些为使用能量提供“操作流程”的序列；而在其他生物体的细胞中，也存在同样功能的脱氧核糖核酸序列，尽管它们与人类的截然不同，甚至相隔千秋万代。这种生物学上的遗留代码有时可能是完全未经修改过的，但是通常来说，在进化进程中，生物系统也会被修补、被改变。

站在生命体的宏观角度上来看，这种情况意味着新功能通常是层层叠加在旧功能上的。这种分层有时会导致一些问题。例如，虽然人类现在已经可以双足直立行走了，但是人类的骨骼结构最初却是为了“四足运动”而存在的，也就是说，本来人类是用“用四条腿走路”的。后来，进化女神不断对我们祖先的身体修修补补，最后给我们提供了一个“足够好”的手段，也就是依靠脊柱实现双足直立行走。但是，这个解决方案远远谈不上完美，比如，有许多人都不得不忍受背痛之苦。

这种惊人的复杂性，以及对进化路径的依赖性，又会带来什么结果

呢？在试图理解这些系统时，生物学家是不能依靠美学观念的。正如生物学家史蒂夫·本纳（Steven Benner）所说，进化而成的系统虽然能够有效地运行下去，但它看上去却不一定“很美”。[8]这样的系统可能很符合功利主义的需求，当然，有些人有时会把“功效”与“美”混为一谈，但是一般来说，它远远谈不上优雅或简单。本纳还进一步指出，真实的脱氧核糖核酸结构绝对算不上美，不过艺术家通过抽象手法呈现给公众的脱氧核糖核酸形象却很美。生物系统是进化而来的“足够好”的系统，而不是设计出来的精致优雅的系统。换句话说，生物系统是拼凑而成的系统。

与技术系统一样，生物系统的进化也会留下遗留代码。[9]我曾在美国中西部地区居住过，当时我家后院种有许多皂荚树，树上密密麻麻地长着很多刺，那些刺又长又大，很是危险。[10]这些刺似乎没有任何功能，除了让我担心会被刺伤之外。皂荚树为什么会长刺？一个解释是，有了这些刺，这些树就可以“确保”叶子不会被猛犸象，或者其他北美巨型食草动物吃掉。当然，那些动物都已灭绝了。随着这些巨型动物的灭绝，对种子荚遗传基因进行编码的信息也势必会出现进化，当然，“长刺”这个信息到现在还没有消失。

虽然在这个问题上仍有很多争论，但许多科学家都认为，在生命体的基因组中存在着许多“多余物质”，这些物质也被称为“垃圾脱氧核糖核酸”片段。它们在进化进程中被积累起来并留存至今，却没有任何特定的生物学功能。许多复杂的技术系统，例如软件系统等，都包含着这类不会再被使用的功能，这些功能甚至可能已完全过时。与此类似，在许多生物系统中也存在不少这样的退化性状，它们原有的功能已不复存在。

对于生物系统和技术系统的相似之处，我们无法完美地一一对应。当然，生物系统处理遗留代码的方式与技术系统是有所不同的，[11]皂荚树最后可能会变得没有刺。如果这个性状确实是无用的，那么长刺对于

皂荚树来说就是在浪费能量。因此，在未来的进化中，在与那些因为不长刺而更能适应环境的树种的竞争中，长刺的皂荚树会败下阵来，从而被淘汰。倘若真是如此，那么我的子孙后代就不用再承受被皂荚树刺伤的风险了，对此他们应该感恩。在大多数技术系统中，这种可能性都是不存在的。尽管遗留代码会让软件程序变得过时且效率低下，但它们却不会被自动清除。

第三，生物系统与技术系统的相似性还可以通过前文所提到过的高度最优化容限模型来进行分析。

某个看上去很强大的技术系统，可能会因为一些很小的干扰就出现灾难性的故障。在生物系统中也会发生同样的事情。例如，总的来说，人类具有很强的环境适应能力，但是人类基因组中的某个微小突变却会导致侏儒症，而且受到影响的两个基因副本是具有致死性的。无论是在生存尺度上，还是在构成材料上，人类以及其他生命体都与粒子加速器或计算机网络有所不同，但是所有这些系统的复杂性和脆弱性却有着深层意义上的相似之处。

总而言之，生物系统与技术系统确实存在着深远的“亲缘”关系，这意味着我们可以从生物学思维中学到很多东西。

## 技术领域的“生物学家”

技术系统变得越来越复杂了，我们对它们的理解也走向了两个极端：要么，只能得到关于系统运行的一般性概念，但对其内部细节印象模糊，甚至一无所知；要么，对系统的若干组成部分有零碎的了解，但并不知道这些组成部分是如何融合在一起的，也不知道该对系统行为作何预期。前者趋于物理学思维，而后者则趋于生物学思维。

面对越来越强的复杂性，许多人选择通过物理学方法，抽象掉细节来获得对系统的一般性认知。例如，在考察一个复杂的社会系统，比如

一家大公司或一个城市时，如果用物理学思维来解释，那么我们所采取的方法可能是：将它的其中一个属性绘制成图表，然后看图表能否很好地拟合为一条特定的数学曲线。这种方法能让我们清楚地看到系统内部所发生的事情，至少可以提供相关的线索。但是，这个系统及其行为之所以能很好地与某条曲线相拟合，原因可能有很多。在这种情况下，你得到的只会是更多的问题，而不是答案。理解复杂系统通常不适合采用直接的、大规模的抽象化方法，因为这些系统实在太混乱、太庞杂了。

因此，注重细节、强调多样性的生物学思维，为理解杂乱的进化系统提供了一个至关重要的视角。只有通过大量的刺激和检验，复杂的进化系统才有可能被完全理解。因此，生物学家，特别是野外生物学家，在研究生命体的复杂性和多样性时，都必定会考虑它们的进化轨迹。这种方法特别适合用来理解技术系统。野外生物学家经常要扮演博物学家的角色，非常注意收集和记录在各地发现的各种事物，并加以分类和编目。除此之外，在面对一个极其复杂的生态系统时，野外生物学家也不会即刻就想要彻底理解它。他们很清楚，一次只能研究这个系统的一小部分，而且即使是针对某一部分的研究，也未必会完美无缺。例如，他们只研究少数物种之间的相互作用，鲜少会去考察某个地区的完整的物种网络。野外生物学家对自己的判断非常坚定，而且明白，在任何时刻都只能观察周遭复杂情况的一个片段。

与此类似，在面对一团乱麻般的技术系统，譬如一个软件、某个国家的法律体系，或是整个互联网时，如果我们硬要将物理学思维中的优雅和简洁附加在其整体性之上，那么我们就不可能走太远。要想从真正意义上理解技术系统，并对其行为做出有效的预测，我们就需要成为技术领域的野外生物学家[12]。

这对我们意味着什么？在思考一个系统的不同交互层级时，我们要记住，那些看上去毫不起眼的底层细节有可能会升至顶层，从而变得对整个系统至关重要。我们需要技术领域的野外生物学家，需要他们来对

复杂系统的细节和各个组成部分及其失败和错误进行编目和研究。这种生物学思维不仅可以带来新的思想，而且完全有可能成为我们探索互联性越来越强，而可理解性越来越弱的技术世界的主要路径。

如前所述，我们可以在技术系统的错误中学习，就像生物学家在遗传错误中进步一样。但是，生物学家所做的并不仅限于此。为了更好地向生物学家靠拢，我们必须更加深入地了解生物学的一般研究方法。

近年来，遗传学领域的一项重大突破是核糖核酸干扰技术（RNA interference）。核糖核酸是脱氧核糖核酸的“表亲”，在细胞中生成，并服务于多种目的。这项技术的核心是，利用核糖核酸的小片段终止某些蛋白质的合成。这就是说，在构建出正确的核糖核酸文本之后，你就可以有效地关闭某些基因了。

那么，这个机制是怎么被发现的呢？最初，一家初创的生物技术公司的遗传学家们，试图用基因工程方法培育颜色更深的紫色矮牵牛花。[13]他们在识别出了矮牵牛花中负责显示紫色的基因后，便设想再加入另一个基因拷贝，以使花朵颜色更浓艳。然而，当他们真的这样做了之后，得到的却是一朵白花，这与他们的设想背道而驰。新加入的遗传信息不但未能让紫色矮牵牛花变得更浓艳，反而使它完全失去了紫色。不过，研究者们既没有因这个意想不到的结果而感到气馁，也没有忽视这个结果，他们敏感地意识到，研究需要继续。最终，核糖核酸干扰技术诞生了。

事实上，这种事情不仅常见于生物领域，在其他科学领域也经常发生。美国著名作家、生化学家艾萨克·阿西莫夫（Isaac Asimov）曾指出：“在科学研究中最激动人心的短语，也是预示着新发现的短语，并不是‘Eureka！’（有了！找到了！），而是‘That's funny…’（有趣的是……）。”[14]生物化学家亚历山大·弗莱明（Alexander Fleming）在培养皿中看到了一些预料之外的奇怪东西，他没有放过它们，于是发现了青霉素。原子核也是科学家们在检验一个

实验的意外结果时发现的，他们原本是想用一片薄薄的金箔映射放射性粒子。科学家们没有对意外结果视而不见，反而通过仔细观察，得出了对原子结构的全新理解。如果借鉴博物学思维，对那些看似不合理的少数特例进行收集和编目，并养成习惯，我们就能对正在研究的事物拥有全新的认识。

当然，仅仅被动等待意外发生，然后加以观察，是远远不够的。生物学家还会积极地采取行动，主动将意外事件注入系统，然后观察系统的反应。在试图培养某种特定类型的细菌，例如某个可能会生成特定化学物质的细菌变体时，他们会采用一种通常被称为“诱变”（mutagenesis）的方法。顾名思义，诱变就是通过积极的尝试来诱导产生突变。例如，对生命体进行辐射，或让它们接触有毒的化学物质等。

虽然这种做法听起来会令人稍感不适，但它绝不是无的放矢。当我们预测不出某个复杂系统可能会产生何种反应时，或者无法确定基因组中的哪些变化可能会产生预期的效果时，我们通常需要利用一定的随机性来找出系统的行为倾向。从根本上说，这些系统都是非常复杂的高度非线性系统，所以我们不得不“借用”自然进化进程中的修补措施，来探索它们的运行方式。

在制药公司研发新药的过程中，也可以总结出类似的工作原理。在研制新药时，有一种方法是：对一种化学物质的多种变体进行筛选和试验，以找到符合疗效的化合物。[15]虽然还无法完全理解化学机制的“真相”，但这样的试验确实帮助我们找到了一些不可多得又行之有效的药物，并让我们对人体有了更深刻的理解，尽管这种理解有时候是间接的、模糊的。为了能更多地了解并筛选出可以合成有效药物的分子结构，我们必须不停地“拨弄”系统。制药业的经验已经证明，这种方法不仅有效，而且是科学发现所需要的修补方法之一。

因此，要将生物学思维应用到对技术系统的研究中，我们就必须认

识到，“修修补补”是构建系统和理解系统的一种重要方法。正如斯图尔特·布兰德在讨论遗留系统时所指出的那样：“要从遗留系统中汲取新功能，不是简单地下一个命令就能实现的，而要通过一系列谨慎的实验，在‘运气’的帮助下，才有可能得到我们想要的结果。”[16]这就是技术领域的生物学方法。上一章中提到过奈飞公司的“混沌猴”，本质上即是一种诱变软件，旨在通过引入故障来更深刻地理解系统，并提高其稳定性。对于所有故障和问题，包括人为引入的，我们都必须记录下来，并进行仔细分析，以更好地理解大型系统的运行情况。

这种生物学思维还可以帮助我们更好地理解天灾人祸。比如说，我们需要借鉴生物学家对癌症的观点：在某些细胞长成肿瘤后，不能简单地说，这是某件事情出了问题。癌症其实是诸多因素和多种生物反应累积所致，而且在这些因素和反应之间还存在复杂的相互作用。癌症是身体出现的大规模故障，这种故障足以致命。与此类似，对于技术系统，我们也要认识到，问题以及系统对问题所做出的各种反应，在积累到一定程度时，可能会导致同样的级联式失败。这就意味着，我们必须像生物学家那样思考问题。在一座核电站中，不断累积的细小问题最终会引发严重问题。例如，在三哩岛核电站事故中，其实有多个独立因素都与该核电站的部分熔毁有关联。[17]同样，我们可以用这种方法将导致金融市场崩溃的各种因素之间的相互作用识别出来。

令人高兴的是，我们为理解技术系统而做出的努力，已经得到技术系统自身的帮助了。在今天，有很多计算工具都可以帮助我们发现系统中的意外结果。这些计算工具皆来自“新颖性检测”领域。也就是说，现在的机器已经可以与技术领域的野外生物学家和博物学家合作了，它们可以帮助我们更好地理解，至少是部分理解技术系统。

## 当物理学遇见生物学

如前所述，技术具有诸多“生物学性质”。它不仅“笨拙”，而且

会在进化进程中因为修补而生长变化，同时还拥有许多繁杂的细节。那么，这是不是意味着，我们应该放弃探寻复杂性背后的规律呢？绝对不是！在理解技术系统时，物理学思维也能发挥重要作用。

我们在试图理解一个复杂系统时，必须先确定以何种分辨率来考察它，或者说，必须先确定所考察的细节涉及哪个层级。在我们所关注的层级上，细节的精准度如何？应该关注生命体细胞中的单个酶分子，还是身体内部的器官和血管？应该关注通过电路传播的二进制信号，还是着重测试计算机程序的整体结构和功能？在更高的维度上，我们是否只需要关注计算机网络的一般属性，而不用关注组成这个结构的单个机器及其决策？

这些问题并不总是很容易回答。有时，我们必须向物理学靠拢，通过抽象掉部分细节来理解整个系统；有时，细节又特别重要，譬如我们在前文中讨论过的罕见词和边界情况，在这种情况下，我们必须更多地依赖生物学思维。

然而，在很多情况下，不同水平的分辨率是会产生冲突的。镰状细胞性贫血是一种相当严重的全身性疾病，但它的致病因素仅仅只是脱氧核糖核酸中单个碱基对的一个微小变化。发生于2003年夏天的俄亥俄州电网崩溃事件表明，即使只是一棵树碰到了电线，也能引发级联效应，导致全美电网大面积瘫痪。不同系统之间的联系变得越来越紧密了，在这种情况下，不但不同水平的分辨率会相互交织，而且原本相对独立的各个领域也会越来越多地产生交集。这就意味着，我们必须越来越多地把物理学思维和生物学思维结合起来，在探寻秩序的同时，也不能忽略粗糙的边缘。在将生物学思维与物理学思维结合到一起后，我们便能更加了解周围各种拼凑起来的系统了。尼尔·斯蒂芬森（Neal Stephenson）在其小说《编码宝典》（Cryptonomicon）中借人物之口，详细地描述了希腊神殿的结构。书中人物所说的话与我们在此阐述的思想不谋而合：

然而，正是这种混乱、这种不对称性，才使万神殿更加可信。就像元素周期表和基本粒子系谱一样，也像你通过尸体解剖看到的人体结构一样，它除了呈现出足够可靠的、我们的大脑能够分析的模式之外，还潜藏着一种不规则性。这种不规则性表明，“有机来源”是存在的。例如，这里的太阳神和月亮女神是明确对称的；这里还有赫拉（Hera），她并不在这种对称性中扮演角色，单纯只是一个女神；然后还有狄俄尼索斯（Dionysus），他甚至不完全是神，而是半人半神，无论如何，他也进入了万神殿，与其他众神一起坐在了奥林匹斯山上。这就好比，你走进最高法院后却发现一个马戏团小丑坐在了一群大法官中间。[18]

我们对身边的各种系统审视得越是仔细，就越能看清生物学与物理学之间的平衡关系。在生态系统中，在我们每天依赖的混沌技术中，皆是如此；在希腊神话中，在讲给自己听的故事中，亦复如是。

事实上，“讲故事”是一个非常好的方法，能够让我们融汇生物学思维和物理学思维。有些故事就像精心设计制造的机器，毫无赘述，所有情节都非常合理。这方面的例子最经典的莫过于“契诃夫之枪”原则，这个原则源于伟大的剧作家安东·契诃夫（Anton Chekhov）的创作理念：故事中引入的任何元素都必须与剧情有关。例如，在第一场中出现过的一支上了膛的枪，在第三场中必须开一次火。

当然，也有这样一类故事：感情丰富，铺陈繁多，所出现的元素不一定会推动剧情发展。例如，荷马在《伊利亚特》中叙述的入侵伊利亚特的希腊战船；克莱默（Kramer）在《宋飞正传》（Seinfeld）中提到的从未露过面的宋飞的朋友鲍勃·萨卡马诺（Bob Sacamano）、洛米兹（Lomez）、科尔基·拉米雷斯（Corky Ramirez）等。虽然不是推动故事情节发展的关键元素，但它们同样很重要。[19]这就是与物理学肩并肩的生物学，当我们讲故事时，在增添故事丰富性方面，两者皆不可或缺。

特效领域有一个耐人寻味的术语：“小东西”（Greeblies）[20]。每当听到这个术语，我就会想起小精灵，或者伶牙俐齿的小丑。“小东西”指的是被添加到某个场景中或某个对象上的小物件，目的是让场景或对象看起来更可信。这就好比，你制造了一艘未来主义风格的飞船，那么总不可能让它完全由一些转角和若干光滑的面板组成吧；它还需要加上出入口、通风口、各种装饰物、管道、隆起物，以及凹凸槽等。《银河战星》（Battlestar Galactica）或《星球大战》等影片中的宇宙飞船，之所以拥有强烈的吸引力和视觉冲击力，就是因为它们都拥有这种“莫名其妙”的复杂性。

对于为什么要不断加入小物件，著名数学家、“分形”的创造者本华·曼德博（Benoit Mandelbrot）的一段名言给了我们很好的解释：“为什么有人会说那些基本的几何形状是‘冷冰冰的’和‘干巴巴的’？其中一个原因就在于，它们无法被用来描述云、山、海岸线或树的形状。云不是球形；山不是圆锥体；海岸线不是圆形；树皮不是光滑的平面；雷电也不是按直线行进的。”[21]

技术系统也是如此。一旦嵌入了真实世界，技术系统就失去了如直线、三角形般纯净的逻辑结构；相反，随着时间流逝，它将因吸积而充满各种杂项，就像生物系统中因积累而出现的“进化大杂烩”一样。故事要以细节和复杂性为基础，系统也一样。事实上，我们之所以会认为某些东西更加“现实”，正是因为它们是复杂的，充满了细微的“锯齿”和各种细节，尽管这些“锯齿”和细节常常是我们一时无法理解的。说到底，我们需要这种混乱，需要这些小物件，即使它们在分辨率水平较低的解决方案中会被全部抽象掉。

生物学思维必须与物理学思维“和谐共处”。不妨回想一下博尔赫斯的短篇小说《博闻强记的富内斯》，主人公富内斯背负着完美记忆的重担。[22]虽然许多人可能会像富内斯一样，将这种能力视为一份天赐的礼物，但事实并非如此。我们可以从小说中得知，对富内斯来说，几乎

每一个细节、每一种观点都会形成新的记忆，继而形成新的记忆类别或类型。当他凝视一只狗的时候，他无法像普通人那般看待狗；每时每刻，他都是在用某个特定视角看待一只特定的狗，只要狗，或他自己动了一下，那么他的大脑就会生出一个关于狗的全新记忆。富内斯没有办法将所有细节统一到一起，并形成某个一般性概念，因为他的记忆过于细节化。换句话说，他无法形成任何抽象的概念，因为他对庞杂现象的容纳度实在太高了。

事实上，生物学家的最终目标同样是创建模型，并识别出规律，只不过这个目标所涉及的范围可能较小。在面对一项复杂技术时，我们需要从野外生物学家的角度出发，围绕其边缘进行各种各样的实验，看看它会做出怎样的应对。当然，在这样做的时候，我们的最终目标仍然是求得某种程度上的一般化。这种方法并不鲜见。很多人在玩诸如《我的世界》（Minecraft）这类开放式游戏时，所采用的便是这种方法。在这类游戏中，你先要收集大量与虚拟世界有关的信息，例如，可以做什么，不能做什么，什么东西会“杀死”你，怎样才能生存下去，等等。然后，你需要构建一个比较小的心理模型，接着在较大范围内求得一般化，当然，相对于整个世界，那仍然只是小范围的一般化。

又例如，你正在使用一个很高级的计算机软件，比如说一个庞大而繁杂的文字编辑软件，然后你发现文档尾注的编号变得凌乱了，一般来说，你并不会因此而感到惊慌失措。你会先看看到底出了什么问题？有几个尾注的编号重复了？它们还能关联到文档中的正确位置吗？通过这种修修补补的生物学技巧，你会对这个复杂系统的细节了然于胸。只要真正地像技术领域的生物学家那样行事，我们就能做到，在对系统各个部分进行仔细研究的同时，还能清醒地认识到它们只是一个更大的、高度互联的整体的组成部分。

接下来，我们的讨论将转向另一个领域，其间的知识可以帮助我们在把握技术系统的同时，实现微妙的平衡。

## 复杂性科学的视角

复杂性科学，是管理和理解复杂系统的一条自然路径。它能够定量地研究庞大且复杂的互联系统，无论是生物个体，还是生态系统；大到万维网，小到电影演员的合作网络。提到后者，我们自然会想到凯文·贝肯定律，也就是六度分隔理论。复杂性科学是一个发展迅猛的、令人振奋的研究领域，我自己也涉足其中。它利用各种强大的思想和数学框架，探寻着复杂系统的模式和含义。它所采用的方法极其广泛，从理解网络结构，到构建拥有大量交互主体的计算机模型，也就是基于主体的模型（agent-based model）。

不过，这种理解系统的方法需要先抽象掉一定数量的混乱因素，并找出规律，以进行清晰的数学推理和分析。如果你正在观察一个巨大的交互网络，比如人们在Twitter上的互动，那么你可以先忽略交互的内容和其他细节，只关注个体之间的关联数量，你会发现，关联数量可以被拟合成一种特定类别的概率曲线，即重尾分布（heavy-tailed distribution）[13]。这个曲线不会告诉我们谁与谁有关联，谁的关联数量很多，谁的关联数量很少，但它足以表明，交互网系统中存在基本的统计规律。

在复杂性科学中，还有一些模型探索的是所谓的渗流（percolation）：流体物质如何基于某种多孔结构进行扩散，例如，石油如何在岩石中渗透。这些模型能够有效地说明，密度的微小变化会导致流体物质在系统中的扩散能力发生巨变，尽管这些模型无法向我们提供诸如“孔隙的确切位置”之类的信息。

毫不夸张地说，复杂性科学中的每个模型都为理解系统提供了独特的视角。事实上，很多时候我们都是在借助简单的数学模型，统一着世间的诸多事物。例如扩散限制凝聚模型（diffusion-limited aggregation model），这个模型表面看来不过是一个抽象的玩物，但

是实际上，它可以被视为多种现实现象的统一模型。

扩散限制凝聚模型简称为DLA模型，其原理是：先在巨型网格的中心设定一个点，然后让其他点在网格上随机移动；在某个点击中初始点之后，那两个点就形成了一个聚合体；接下来，让剩余的点继续移动，直到它们中的某个点又击中聚合体。因为加入聚合体的点越来越多，所以聚合体缓慢地“长大”了。

聚合体最终会变成一个形状怪异的东西，不能被称为“团”（blob），也不能被称为“畸形的圈”，也不是普通的结晶体，而是一种更加“有机”的东西，如同生物中的徒长（spindly growth）[14]。事实上，它的“模样”取决于模型的类型，有时候会像珊瑚，有时候会像闪电，甚至会像城市。由此可见，一个简单的、无足轻重的计算机模型，可以生成非常复杂的图形图像。

这个模型，以及其他一些类似的模型，将一个“悖论”带到了我们面前：我们可以通过非常简单的模型来复制各种复杂系统，当然，具体细节是不一样的。这些小型“玩具模型”具有某种“波特金复杂性”（Potemkin complexity）。“波特金复杂性”是指，某些系统看上去很复杂，但本质上却很简单。例如，由它们复制而成的“大城市”与真正的大城市有所不同；真正的大城市，当前的状态包含着大量以往的决策。由它们复制而成的“珊瑚”也与真正的珊瑚不同，后者的所有细节都会融入它的生长过程。

尽管如此，在很多时候，有这些模型就已足够。对于某些基于特定目的的观察，例如，在试图理解一个实体组织的组织形式时，我们所关注的就是总体形态。对复杂系统的抽象化和简化，不仅可以帮助我们理解系统行为的总体情况，还可以告诉我们系统的反馈机制、互联性，以及对初始条件的深刻依赖性，甚至可以告诉我们系统会对变化做何反应，并在此基础上帮助我们设置边界。利用复杂性科学的相关知识对复杂系统进行高效思考，是当下每个人都应该掌握的技能。不管是专家还

是普通人，只有具备了这样的技能，才能在这个日益复杂的世界中好好地生存下去。因此，教会人们如何把握复杂系统的运行机制，成了一件迫在眉睫的重要任务。

但是，复杂性科学的一般化模型也是具有局限性的。在试图了解不同系统的特殊情况时，简单模型就不够用了。在这一点上，菲利普·鲍尔说得非常到位：“河道网络的互联模式与视网膜神经的互联模式，既有相同的地方，又有不同的地方。将它们统称为分形模式是不够准确的，它们各自的分形维数也是不够精准的。要想充分地理解河道网络，就必须考虑物质的沉积和输送、气象条件的变化、河道底层基岩地质的具体性质和差异等一系列情况；而这样的思考过程与神经细胞网络毫无关系。”[23]

不过，令人欣慰的是，复杂性科学所提供的工具不仅可以用于创建经过简化的抽象模型，还可以为我们探究系统细节提供窗口。也就是说，它可以积极地、严格地筛选出系统的复杂性。举例来说，一些研究者利用软件工程和复杂性科学的某些方法，分析出了《美国法典》这个法律系统的诸多性质。[24]这项研究的分析结果体现出了这个法律系统的很多总体特征：其不同组成部分的复杂程度不同；在总体上存在着一定程度的庞杂性，等等。同时，他们还筛选出了特别有意思或特别错综复杂的一部分法律，比如拜占庭式的、有隐秘含义的法律。在涉及银行义务的那部分条文中，突然跳出了详细说明“公司的权力和义务”的一节，而且内容特别复杂，至少从条件的数量上来说是如此，它的存在就像是软件系统中的“if-then”语句。与此类似，在关于税收的部分，“适格的养老金、利润分配和股票红利计划”一节也特别复杂。这些研究结果显然有助于我们集中关注，并“放大”分析法律体系中最复杂的部分，甚至有可能会起到促进简化的作用。

与野外生物学方法一样，某些复杂性科学方法可能更适合用来了解复杂系统中某个子系统的行为，而不是整个系统的行为。我们还可以利

用复杂性科学的方法来找出异常值，也就是复杂系统中不符合一般规则的部分。在所有这些方法的共同作用下，我们便可以确定系统中哪些部分更值得深入研究，从而更好地了解它们是如何相互作用的。我们必须协调好抽象化的复杂性科学方法与其他方法之间的关系，后者往往可以帮助我们找到“无法平滑嵌入模型”的特定部分和细节。

在这里，我们必须再一次面对物理学思维与生物学思维之间的紧张关系。一方面，我们希望技术世界拥有简单和优雅，希望那些混沌不明的东西消失；但是另一方面，我们必须承认，有些事物既是庞杂的，又是微妙难解的，还会随着时间的推移发展和进化，这便是心智成熟的标志。在成长为独立的个体之后，我们会认识到人际关系的复杂性，认识到人际交往的很多精致微妙之处。与此类似，当我们所处的社会逐渐成熟起来之后，人们必定会认识到社会结构中固有的复杂性和不规则性。复杂性科学，尽管绝非“万妙灵丹”，但确实可以帮助我们实现这种平衡：既能突出需要特别关注的细节，又能为知识范畴和关注程度设定恰当的边界。例如，复杂性科学可以告诉我们，系统为什么很容易就会变得不稳定：如何变得不稳定，以及我们必须特别关注哪些地方。如果一个简单模型已经证明了，某个大型技术网络中的一个微小变化很可能会导致网络彻底崩溃，那么我们绝不能对此保持“无知的幸福感”。

知易行难。想要在物理学思维与生物学思维之间找到平衡点，并非易事。这是一个艰难的探索过程。在漫长的思想史上，人类一直沿着这个方向不懈地探索着。

## 思维方式的进化

事实上，许多先贤都曾试图理解周遭世界。在历史上，爱奥尼亚（Ionia）是最早出现这种严谨思想活动的地区。爱奥尼亚是一批古代城邦国家的总称，这些国家位于爱琴海之畔，也就是现在土耳其的西海岸，古希腊的第一批哲学家就出现在那里。在苏格拉底时代到来之前，

他们已开始潜心思考宇宙的奥妙。这些古希腊哲学家的思想是前苏格兰学派思想的一部分，并且拥有一致的特点：压倒性的简化和一般化。[25]

爱奥尼亚诗人兼哲学家色诺芬尼（Xenophanes）说过两件事，或者说是一件事，这取决于你的理解：世间万物，皆源于“土”和“水”；归根到底源于“土”。无论如何，色诺芬尼是在尝试着用一个或两个基本原则来解释世界的复杂性。

与此同时，另一位爱奥尼亚哲学家泰勒斯（Thales）则认为，万物皆由“水”组成；还有一位哲学家认为，一切皆来源于“火”。这样看来，“地球超人”（Captain Planet）[15]应该会喜爱这些爱奥尼亚哲学家。

米利都的阿那克西曼德（Anaximander of Miletus）是又一位爱奥尼亚哲学家，在他看来，一切事物都源于单一的、无限的第一原则。阿那克西曼德还提出了其他许多颇为精确的理论，例如，动物来自“被太阳蒸发的水分”。[26]他还假设，地球正被一个相当于自身体积28倍大的圆圈包围着，并由此推导出了不少天文学理论。

不过，总的来说，这些哲学家都倾向于：只需少数几种基本材料和少数几个基本原则就可以解释整个世界。或者说，他们几乎都坚持了古希腊人关于kosmos（宇宙）的基本观念。与现代英语中的cosmos（宇宙）有所不同，希腊人不仅用这个词来指代宇宙及万物所依赖的整体系统，还用这个词来形容宇宙的优雅和美丽，以及终极秩序。也就是说，大自然的优雅和保持秩序的物理定律，其实都已隐含在这个表达自然的词汇之中了。除了“宇宙”这个术语之外，这些哲学家还常常使用另一个术语：“本原”（arche），其含义与作为宇宙基础的指导规则，也就是宇宙法则很是接近。很显然，这是一种渴望统一世界秩序的愿景。[27]

离开古希腊，来到现代社会的黎明时代，我们会发现思想家们的观点发生了一些转变，他们更加关注奇怪的、无法解释的事物和现象。我

们看到了“杂项”。根据菲利普·鲍尔的研究，古代和中世纪的学者对异常现象和古怪行为的兴趣普遍不高，他们感兴趣的是确认已知的东西。[28]但是，对于那些致力于建立“珍奇博物馆”，站在现代社会入口，处于风口浪尖上的开创者们来说，那些异常现象和古怪行为才是这个世界上最令人激动的事物。科学史学家洛琳·达斯顿（Lorraine Daston）认为，在科学发展的早期阶段，也就是在费尔法克斯时代到来之前，科学家们都醉心于“奇怪的事实”。[29]

关于这一点，可以来看看达斯顿所引述的，于17世纪发表的一系列科学论文的标题：《怪物头上的可见物》《背上有洞的野猪，当它被猎人追赶时，洞就会冒泡》《一位潜水员的讲述：恐怖的雷电奇闻》……从这些标题中，我们不但可以发现不少怪异的信息，也不难判断出当时的科学家喜欢收集奇异事物。他们渴望记录和解释自然世界中那些最意外和最古怪之处。正如达斯顿所指出的那样：“第一批科学事实是很难处理的，并不是因为它们很稳定，我们无法对它们视而不见，而是因为它们太怪异，我们无法将它们归于某个理论。”[30]

很难想象，现代科学作为收集新奇事物的手段之一，却没有办法解释那些新奇的东西，但这就是当时的科学，存在于生活的方方面面。当今时代，我们不再把科学视为收集古怪事物的一种活动了，更不会认为科学就是对这些事物的解释。然而在历史上，科学的确曾是这样的活动，而且直到今天，在很多方面仍有这样的“遗风”。当我们尝试认真面对环境的畸变和异常现象时，我们通常能够学到更多的东西。例如，物理学实验中的一个异常现象最终揭开了原子核结构的面纱；核糖核酸干扰技术的诞生也可以归因于一次意外。

这是一种融合了物理学思维的生物学思维。我们必须把这种思维方式引入到技术领域中。对于这项工作，从一定程度上来说，人人都可以参与其中。此外，我们还需要技术领域的野外生物学家，需要他们在复杂的技术大厦中挖掘“意外之物”，并与难以理解的事物“做游戏”。

那么，这些人会是谁？在当下这个专业化时代，我们需要一些通才，他们有能力识别并牢牢抓住系统的各种细节，以及未被理解的边界情况和关键节点。

## 我们需要通才

2009年，一个科学家小组利用在线科学论文网站，对用户的数亿次互动进行了分析，希望能辨析出读者的“点击流”，也就是读者从一篇论文出发，去往下一篇论文的路径。[31]研究数据最终揭示了人们从一个主题领域移动到另一个主题领域的行为模式。科学家甚至还生成了这些主题领域的互联“地图”，图像非常漂亮。在这个“地图”中，护理学位于心理学和教育学附近；有机化学是物理化学和分析化学之间的纽带；无论是经济学还是社会学，都和法学有关联；音乐则显得有点“特立独行”。

当然，这样的互联“地图”有些过于简单了。实际上，音乐也会涉及物理学和心理学的概念，而经济学则会大量地从数学中汲取营养。不过，无论如何，分析“点击流”确实是探索思想互联性的有效途径之一。为了理解我们自己所构建的、越来越复杂的系统，专业化分工仍在继续深化，但是互联“地图”提醒我们，没有哪个领域可以孤立存在，每一个领域都是这个广泛互联的系统的一部分。

因为各种系统会以多种不同的方式相互关联，所以人类越来越需要将某个知识领域与另一个知识领域联系起来。例如，在编写游戏《危险边缘》（Jeopardy!）的计算机程序时，你需要运用从语言学到计算机硬件领域的相关知识。在这种情况下，仅靠专业化是行不通的，我们还需要依赖相当广泛的知识。但是，如前所述，我们的知识体量很快就会触及上限。是的，人类大脑的生理特性决定了人们无法掌握所有知识。

为了克服这种困难，我们需要培养通才。在这里，通才是指既能看

到“土地的地形地貌”，即拥有抽象的物理学思维；又能在尚未理解整个系统的情况下便懂得欣赏系统细节，也就是拥有复杂的生物学思维的人。毫无疑问，在构建复杂系统时，通才是最适合成为博物学家和野外生物学家的人，而且必不可少。他们可以从一个片段跳到另一个片段，检查出那些没有意义的部分，并在巨大的技术系统中挖掘出正在发生事件的线索。

或许有人会问：既然知识量增长得如此迅猛，那么当今世界是否还有通才存在？

当今世界，通才当然是存在的，只不过想要成长起来却非常艰难。要培养出能够很好地扮演上述角色的通才，第一步是要先培养出一大批“T型人才”。这个术语最早出现在计算机教育领域，[32]而后泛指在某个领域内既拥有高深的专业知识，也就是拥有T型中的“主干”那一竖；又拥有广泛知识，也就是拥有T型中的那一横的人。

那么，这种类型的人应该是什么样的呢？数据科学家正是T型人才的典范。数据科学家的职责是利用计算机科学和统计工具挖掘大型数据集中的隐藏含义。这项工作与具体学科没什么关系。为了很好地完成工作，数据科学家必须了解多个领域的专业知识。我们在应用数学家身上也看到了类似的特征，他们要跨越学科界限，用定量工具找出各学科的共同之处。这便是通才的模样。

通才还可能出现在咨询和图书编辑等行业中，你甚至可以在风险投资界看到他们的身影，在那里有很多人不但拥有多领域的丰富知识，而且还懂得如何有效利用这些专业知识。如果有人想同时在外层空间、3D印刷、农业技术和科学探索工具等领域有所斩获，那么他必须得拥有通才的潜质。

通才既拥有专业知识，又乐于探索诸多不同的领域。但是，要想培养出T型人才并不是一件简单的事。他们先要将专业化与普遍性结合起来，才能着手去解决那些日益复杂的问题。

在当今社会中，专业化人才深受就业市场的追捧，这让培养T型人才这件事难上加难。[33]在我看来，在未来相当长的一段时期内，我们都很难在学术界找到这类才人。正如商学教授大卫·蒂斯（David Teece）所指出的那样，只有在文艺复兴时期，成为“文艺复兴人”才那么容易。但是，从培养“女童子军”这件事中，我们看到了一丝曙光。[34]

女童子军组织曾颁发过一种非常有意思的荣誉徽章：“业余家”徽章（Dabbler badge）。此举是为了对那些想在每一件事情上都做好的女童子军们给予认可与鼓励。这种做法是值得推广的。如果你不想只学制陶或摄影，而想多学一些有用的东西，并且你样样都学得不错，那么“业余家”徽章即是为你量身定做。我的母亲曾经告诉我，她当年在参加女童子军后，获得的唯一一个徽章就是“业余家”徽章。

或许，现在是时候推广“业余家”徽章了。对那些致力于了解各领域思想和技术的人，我们应该就其能力和贡献给予认可和鼓励，无论是在其小学阶段，还是在其职业生涯的最后阶段。他们可以帮助行业专家们将一个领域的专业知识“翻译”到另一个领域中去，以此化解“隔行如隔山”的困境，抵御“不可理解性的洪流”。这些“业余家”可以为巨型系统的构建工作带来生物学思维：找出系统中可能会发生的故障，对这些系统做出深入的理解，并对系统中那些奇异和惊人的组成部分进行分类和编目。

事实上，最令人费解之处往往最能让通才感到如鱼得水。在那些令人费解的地方，系统是如此复杂、如此紧密地相互联系着，以致我们只能尽力将“杂项”记录下来。这就意味着，通才的培养不仅在于学习已知知识，更在于探索未知领域，寻找前人所不及的新方法。而这又意味着，通才不仅要掌握严谨的人文科学思维，还要拥有数学思维和逻辑推理能力，甚至还要学会数据可视化技能，等等。在这里，或许我们可以用中世纪早期的英国小说《希尔德》（Hild）中的主人公来作类

比。[35]希尔德天生聪慧，因擅长织造挂毯而为人熟知。说起她操作织机、飞速织毯的技巧，人们都形容她“眼睛一转就能创造出一个新花样”。对于她的智慧，人们的评价亦复如是。这种建模思维（pattern-making mind）的本质特征是：能够创建联系，并看清互联网络。正是因为拥有这种能力，希尔德才得以游刃有余地周旋在种种地缘政治的阴谋中。

当然，如果让通才“单独行事”，那么他们够发挥的作用也相当有限。只有在与专家一起工作时，通才才能发挥出最大的作用：他们能够在翻译和交流的过程中为专家提供很大帮助；或者对专家的工作进行补充，而这种互补性非常重要。由此可见，在企业或组织中，为每个大型项目配备通才，或者流动性的通才小组，是很有必要的。通才可以为整个项目提供必要的背景知识和价值。必要的时候，企业还可以设立一个通才部门，对外服务，与专业公司合作，形成互补优势。

通才并不仅仅是T型人才。T型人才只是比一般的专业人士拥有更广的知识面；但作为通才，还必须有意识地将多个有实质性区别的领域联系在一起，尽管这样做有可能会失败。最优秀的通才，能够将如编年体般的历史细节与建模思维完美地结合起来。拥有建模思维能力的人，不会是那些只擅长抽象化和一般化的人，而是那些善于创建联系和类比的人。在高度复杂的系统中，对各部分相互关系和交互作用的洞察是非常重要的。因而，拥有建模思维的人，通过直觉和生物学思维相结合的方法，便能够理解，至少是部分理解这些系统。

我是国际赏云协会（The Cloud Appreciation Society）的成员。在我眼中，尽管万里无云的碧空也很美丽，但如果能形成一定的对比，或许会更令人激赏。也就是说，天空需要云。在我第一次告诉我太太，我是赏云协会成员的时候，她以为那是一个稀奇古怪的组织。于是，我给她看了很多“证据”：终身会员资格证、小徽章和官方证书等，但这反而让她觉得更加不可思议。然而，我真的认为，云既美丽又

迷人。不过，在这个方面，我的理解并不比任何一个热情的业余爱好者深入多少。

未经专业学习的人，或许知道云的种类有很多，却不一定清楚它们之间的区别。你可能会说，有暴风雨来临之前的云、蓬松的云、色彩斑驳的云等，然而这是相当幼稚的分类方法，不能充分反映出云的复杂性和多样性。按照正式的分类和称谓，我们常见的云包括：积云、卷云、雨层云、积雨云等；还有一些特定的子类型，譬如碎云、陆架云、荚状云等。事实上，对云的科学分类是在近几个世纪里才发展起来的。正如科普读物出版家约翰·普塔克（John Ptak）所说：“云‘逃过’了古往今来最伟大的分类专家亚里士多德的法眼，后来又很少受到其他科学家的关注。[36]而且，20多个世纪以来，没有人对云进行过科学分类，直到……1803年，英国制造化学家、气象学家卢克·霍华德（Luke Howard）公布了他对云的分类。”

发生在天空中的其他现象也激发出了许多美妙的气象术语，比如片状闪电、圣埃尔莫之火（St. Elmo's fire）、球状闪电，不一而足，它们大多都很怪异。就气象科学而言，在预测天气和解释气象方面，都已取得了很大进展；但是纵观历史，新奇天象层出不穷，这又为我们思考“大气如何影响天气”提供了新的思路。

云和其他大气现象，虽然不是人类所制造，但我们仍然可以从对待它们的方式中，学到一些对待复杂技术系统的思维方式。当我们不理解某个现象或某个系统或心生畏惧时，不应该避之若浼。有些事情即使暂时无从解释，但也应有一席之地；虽然处在尚待理解的范畴，但也可成为深入研究的楔子。

在面对某个完全无法理解的事物时，可以暂且先关注该系统中的细节，尝试着去理解整体中的某些特定部分。说到底，我们在研究系统时所采用的生物学方法，其实就是迭代和修补。也就是说，细节和意外情况不但能够帮助我们更加深入地理解系统，还能不断提高我们的洞察

力。

在复杂难解的事物身上，自有神奇曼妙之处。那个闪耀着炫目光芒的技术网络复杂得令人难以置信，因为其各个组成部分之间拥有着紧密的联系和相互作用。许多时候，我们无法了解它的每一个部分，也无法完全理解它的整体性，不过，能对它有个不完美的把握，或许已经足够。我们可以与技术携手，谦卑前行。

# OVERCOM PLICATED

## Technology at the limits of comprehension

认识复杂系统的正确态度是：对于难以理解的事物，要努力克服我们的无知；一旦理解了某个事物，也不会认为它是理所当然的。谦卑之心，加上迭代的生物学思维，就是洞悉复杂世界的正确方式。

06

**生物学思维是理解复杂世界的一把金钥匙**

认识复杂系统的正确态度是：对于难以理解的事物，要努力克服我们的无知；一旦理解了某个事物，也不会认为它是理所当然的。谦卑之心，加上迭代的生物学思维，就是洞悉复杂世界的正确方式。

摩西·迈蒙尼德（Moses Maimonides）在哲学史上是一个独具特色的人物。他生活在12世纪，从小就接受了成为医生和拉比的训练，后来在埃及做了一名宫廷医生。他不仅编纂了犹太法律，还对当时的科学思想进行了融合与创新。他的崇高地位在犹太传统教义中得到了充分肯定：“从摩西到摩西，从来没有出现过像摩西这样的人物。”迈蒙尼德还著有一本哲学教科书，名为《迷途指津》（The Guide of the Perplexed）。这是一部伟大的作品，将犹太教思想与亚里士多德的哲学理论融合在了一起。这个尝试很大胆，毕竟在那个时代，亚里士多德的哲学理论被公认为最先进的哲学思想。无论在犹太人当中，还是在非

犹太人当中，《迷途指津》都拥有非常多的读者。

那么，我们可以从这位学者身上学到什么呢？在好奇心的驱使下，迈蒙尼德几乎研究了从营养学、天文学到政治学的所有学科。他认识到，尽管人类拥有非凡的智力，但是仅靠好奇心是无法理解世间万物的；至少，在涉及“无限”时，人类的思维能力是必定不够用的。尽管迈蒙尼德的观念在很多方面都领先于时代，但他还是受到了时代的限制。他认为唯一可以理解世间万物的只有“上帝”。当然，在迈蒙尼德的字典里，“上帝”是比“天上的伟人”更抽象、更具哲学意义的概念；而“天上的伟人”则是人对“神”的共同认知。迈蒙尼德坚定地认为，人类的思维能力具有局限性：“……人的智力，毫无疑问，是有限的；人死去后，智力也就消失了。因此，人们都明白，世间必然存在着某些自己无法理解的事物。而且，他们还发现自己的灵魂并不渴望这种知识，因为他们已经意识到了这种知识的不可理解性，意识到了获取这种知识的途径是不存在的。”[1]

几百年后，人类的思维方式已经缓慢而稳定地实现了转变。而今，我们用“科学革命”这个术语来定义这种转变。随着现代科学的发展，科学家开创了天体物理学和化学等新领域，这让我们认识到某些局限性是可以克服的。当然，这种转变绝不是突然发生的，毕竟像牛顿这样的杰出人物，也将一生中大部分的时间和精力花在了对炼金术和中世纪神秘主义的追求上。无论如何，总的来说，科学观念还发生了明显的转变，人们开始认为，人类的大脑原则上可以理解它想要理解的一切。

这种信心渐渐渗透到了我们对待科学的态度当中，而到了当代，这种信心已经升华为一种必胜的信念：只要足够努力，随着时间的推移，我们定能理解一切。事实上，在探索宇宙和寻找答案的道路上，我们确实已经走了很久了。

在科学方面，我们再次向人类极限发起了冲击。现在看来，想要回答某些关于宇宙本质的问题，恐怕不得不去做一些“不可能”的事，并

且要以超光速旅行。另外，对量子世界的研究似乎也正在接近极限。换句话说，我们正在探索的问题是目前还无法回答的。用生理学家、生物化学家、群体遗传学家J．B．S.霍尔丹（J．B．S．Haldane）的话来说：“现在，我所怀疑的是，宇宙不仅比我们想象的要怪异，而且也比我们能想象到的要怪异。”[2]虽然不应该放弃探索，但是答案可能会停留在我们的认知范畴内，而我们的认知能力正越来越多地受到限制。[3]

对于自己所创造的事物而言，情况也是如此。不难看到，我们对计算、运输、医疗设备，以及其他许多技术的理解都相当有限。所以，当我们继续构建无从理解的事物时，应该重新审视一下人类早期的思想。在那个时代，人们普遍认为，有些知识是任何人都无法获得的。

那么，在这种情况下，我们应该作何反应呢？我们在本书导论中提到过，随着技术变得越来越复杂，我们渐渐失去了理解它们的能力，在这种情况下，我们的反应会趋于两个极端：要么恐惧，要么崇敬。

一方面，在面对异常复杂的科技系统时，一些人会被它的强大和复杂压倒，并因未知而感到恐惧；另一方面，在面对科技所展现出的美丽与力量时，另一些人则会像对待宗教信仰一般表现出无比崇敬。视频游戏设计师兼作家伊恩·博格斯特（Ian Bogost）甚至认为，用“上帝”这个词取代“算法”这个术语，一点也不会改变“技术”一词在流行语中的风评。[4]

虽然技术推动了社会发展，但它的构建过程并不完美。[5]技术是拼凑而成的系统，这就是说，技术是在时间的长河中，由许多碎片积累、层叠而成的系统。因此，尽管技术拥有令人振奋的力量，但它并不值得我们毫无保留地卑躬屈膝、赞叹不止。当然，我们也不必因技术的存在而感觉自身受到了威胁。

恐惧和崇敬都不是生产性的反应。这两种反应都阻断了我们提出质疑的机会，磨灭了理解技术系统的可能性。虽然在处理中高难度的事情时，我们能做到谨慎有加；但在面对完全无法理解的事情时，我们常常

会受制于恐惧，并停下探究的步伐，而且多多少少会忘记自身责任，忘记我们应该尽力去理解能够理解的事情。对技术的崇拜也会导致同样的问题。如果我们对一种算法或技术的评价言过其实，无论是说它“美轮美奂”，还是说它“刻骨铭心”，我们都会因此而失去质疑的机会。

幸运的是，在上述两种极端反应之间，我们还有第三条路可以走，那就是谦卑。我们必须保持谦卑，而不应盲目崇拜；必须满怀好奇，而不应心怀恐惧。对此，计算机科学家迪科斯彻甚至说“卑微的程序员”[6]必须尊重“人类心智的内在局限性”。即使我们能够消除心智上的各种偏差，或者大幅度地增强智力，我们的生命也是有限的。面对这种终极的有限性，我们必须心存谦卑。谦卑既不是桀骜不驯，也不是委曲求全，而是承认自己的局限性，同时不会因这种局限性而陷入无为，也不会将这种局限性奉为神祇。以谦卑之心对待技术系统，有助于我们理解各种各样的系统结构，当然，这种理解仍在我们的认知范畴内。这种对待技术的谦卑态度与生物学思维十分合拍。虽然不断积累的知识让我们对系统有了更多更好的理解，但是这种迭代过程永远都是不完整的。当然，这没有什么关系。

《纽约时报》专栏作家戴维·布鲁克斯（David Brooks）曾经明确指出：“智慧始于认识论中的谦虚。”[7]我们心怀谦卑，并对复杂系统的细节充满浓厚兴趣时，就能做到那些满怀恐惧或崇拜之心的人无法做到的事情。谦卑和兴趣能够帮助我们不断探寻系统背后的奥秘，即使永远无法完全理解系统也没有关系。在许多情况下，部分理解已是我们能达到的最好的效果了。毕竟，聊胜于无。

恐惧和崇敬皆会诱使我们举手投降，放弃对技术系统的努力探索，但我们绝不能这样做，必须继续努力，尽力去了解这些系统。谦卑仅仅意味着我们需要接受以下观点：科学必胜论是错误的。我们永远无法实现完全或完美的理解。只要接受了这一点，我们的灵魂就不会再渴求这样的知识了，正如迈蒙尼德当年所说的那样。而后，我们会变得更

加“通情达理”，不会再为无法理解的东西感到沮丧。当今时代，即使是软件专家也不得不承认，某些计算机错误属于形而上学领域。因此，是时候用谦卑之心来对待技术了。

事实上，在面对复杂的技术系统时，谦卑和包容是非常值得尊重的选择。研究生命基因组的科学家就是这方面的典范。在基因组中充斥着无数无“目的”的片段，它们是在奇异的吸积过程中形成的。在技术系统中也有类似的过程和片段，而科学家却认为这是“光荣的混乱”。[8]不难看出，用“光荣”来形容“混乱”，完全是生物学思维，满含谦卑与钦佩。技术系统是混乱的，是永远无法被完全理解的，但我们必须认识到，这就是“光荣的混乱”。这是一种非常有价值的乐观主义精神：繁杂的细节和不完美的理解都无法压倒我们，反而会令我们兴奋不已。也许它们永远不能为我们带来对整个系统全面且深刻的理解，但那不也很好吗？

约翰·盖尔（John Gall）是一位退休儿科医生，也是畅销书《系统圣经》（The Systems Bible）的作者。这本书原名为《一般系统滑稽论》（General Systemantics），于1975年首次出版，经过修订和扩充后的最新版第3版前段时间面市了。这本书对“如何处理复杂系统”做出了妙趣横生的探索，不过盖尔所指的“系统”具有更加宽泛的意义，涵盖了所有的社会系统和人类所构建的技术系统。书中金句连连，比如，“旧系统的幽灵会不断骚扰新系统”“系统总是会反击”。书里还提到了无意识定理（Unawareness Theorem）：“如果你压根不知道你已经出了问题，又怎么可能会寻求帮助？”[9]盖尔总结的定理以及对系统的分析都非常有见地且非常有趣。而且从这本书的原名出发，我们不难推测出，他的基本结论就是：系统很容易做出一些古怪的行为和出人意料的事情，比如对我们“反咬一口”等，而我们不可能根除这种行为。

透过那些金句，我们可以看到，盖尔提出的许多观点，与我们在本

书中所阐述的思想类似。[10]例如，系统会吸积、会扩张、会发展，并最终“超越人类的评判能力”，做出意想不到的行为。不过在这里，我将只针对其中一部分观点进行讨论。盖尔指出，从手头已有的东西开始构建，要比从头开始更加容易，因为从头来过所导致的问题可能比预期的要多很多。如果不得不从头开始创建一个全新的系统，那么我们要尽可能地控制系统规模。他认为，在减少系统故障方面，某种程度上我们还是可以想到一些办法的。

说到底，盖尔提出的金句和定理可以归结为一点：在面对很难从头开始设计或构建的系统时，务必要保持谦卑。[11]无论这些系统的起源或功能如何，它们最终都会变成笨拙的拼凑而成的系统。同时，在试图理解和控制它们的过程中，我们必须容忍许多混乱之物。只有在充分认识到“系统总会变得复杂”的时候，我们才能更好地构建系统，才能在复杂系统出现各种情况时，不至于被它们“反咬一口”，并且能够应对自如，甚至欢欣鼓舞。

在复杂的技术系统面前，保持谦卑将使我们获得良好的服务。这是一种充满智慧的谦卑，它的关键特征之一来自生物学思维的如下洞见：对于这些复杂系统的内部运行情况，窥一斑而难知全豹；或者说，我们只能在庞大的机器上打开一扇小门。但我们尽力了，这便已足够。

## 不要被表象迷惑

设计师唐纳德·诺曼（Donald Norman）正在将电脑中的文件备份到一台服务器上。他坐在旁边，观察着进度，想看看计算机一步步地都做了些什么。[12]一段时间后，他注意到备份进度已经来到了“样条网格化”（reticulating splines）阶段。这个专业术语看上去很高级，也显得很令人放心——计算机程序肯定真的知道它自己在做什么。但是诺曼没有感到放心，他想知道这到底是什么意思。经过一番研究后，他发现，正如《模拟城市2000》（SimCity 2000）游戏的拥趸们都知道

的那样，这实际上只是一个“圈内笑话”。这款游戏被插入了一些无用的短语，只是看起来有意义罢了。自那以后，这种情况便陆续出现在了各种各样的游戏和其他软件中。

不妨回想一下，上次安装新软件的时候，你真的知道计算机内部发生了什么事情吗？你是否了解各种软件包被存储在硬盘驱动器上的哪些文件夹中？你知道软件安装程序根据计算机硬件和操作系统的不同性能，修改了计算机的哪些信息吗？

不太可能。你眼中只有一个长条状的东西，你只能通过观察它被逐渐填满的过程，来了解安装的进度。它就是进度条。这种小型的界面创新程序，出自计算机科学家布拉德·A.迈尔斯（Brad A. Myers）之手。在读研究生时，迈尔斯开发出了这个程序，并将其称为“进度完成指示器”。[13]进度条给不透明的过程提供了一个小窗口，并以此来抚慰用户。那么，进度条是不是完全准确的？很可能不是。在某些情况下，进度条几乎完全脱离了底层的运行过程。[14]但是在大多数情况下，进度条和其他讨巧的设计，例如描述软件安装过程中所发生之事的文本框一样，可以让人们对一个庞大且复杂的过程投去“放心的一瞥”。

我们在构建用户界面时会越来越多地有意地将复杂性抽象掉，或者，至少让部分复杂性与用户隔离。这种做法使复杂性科学和用户界面设计学这两个领域产生了交集。无论是在电脑中，还是在汽车中，抑或是在电器中，技术系统都会在用户与程序运行之间设置一道屏障。许多人都会选择使用用户界面相当友好的特波税务软件（TurboTax），而不会选择与越来越复杂的税法进行面对面的“博斗”。[15]然而，在这个软件的背后，其实隐藏着一整套非常复杂的法律法规体系，它们是通过大量if语句和处理例外情况的计算机代码来呈现的。

凭借直觉，我们能察觉到系统内部到底发生着什么，尽管这种直觉有时候不甚准确。因此，只要想办法保证这种直觉不消失，就可以消除用户因未知而产生的不安。

我家的第一台电脑是康懋达公司生产的VIC-20。在其代言人、《星际迷航》（Star Trek）的主演威廉·夏特纳（William Shatner）口中，它是“20世纪80年代的奇迹电脑”。我拥有许多有关这款古董级电脑的美好回忆。我曾经用它玩过不少游戏，当时所用的存储设备还是最原始的盒式磁带机。我和我的兄弟玩过很多次的一个游戏是《吃豆人》（Pac-Man），不过，在那款游戏中，“吃豆人”的形象不是一个长着大嘴的黄色馅饼，而是一辆赛车。

我们还玩了不少需要手动输入代码来运行的游戏。在那个年代，虽然你也可以购买现成的VIC-20游戏软件，比如赛车游戏，但是人们玩电脑游戏的主要方式是将杂志上发布的计算机代码输入到电脑里。你想玩一个有趣的滑雪游戏吗？买一本杂志，将上面的计算机代码输入到电脑中，然后自己来运行这个程序，并不需要购买游戏软件。当时，这样的游戏很常见。今非昔比，如今，几乎每个游戏程序都有几十万行代码，可能整本杂志都登不完。

自己输入代码，可以让我们更近距离地接触计算机。我看到了软件中的bug是如何产生的。记忆中，我输入的滑雪游戏的程序代码导致屏幕一侧出现了很多图形乱码，我费了很大心思才把这个错误纠正过来。我还看到了计算机程序的逻辑和内在结构。今天的计算机程序是以成品的形式交付给用户的，相比之下，往日的计算机程序则具有清晰的结构。

20世纪80年代末，我的家人放弃了那台康懋达公司生产的VIC-20，转而使用苹果公司的产品。自那以后，我一直在使用苹果电脑。我们家的第一台苹果电脑，在年幼的我们看来是一个令人难以置信的产品。无论是操作便利的鼠标，还是大量的游戏，都深深吸引着我。我记得其中有一款游戏叫《宇宙渗透》（Cosmic Osmo），它拥有内涵丰富的游戏场景，令人顿生身临其境之感，而且只需点击鼠标即可尽情探索。早期的苹果电脑也会“说话”，能将文本转换为单调的“非人

类”语言，这让我们全家都感到无比兴奋。1984年，史蒂夫·乔布斯在苹果电脑新品发布会上的演讲感人至深，给我留下了难以磨灭的印象。然而，在拥抱苹果电脑这个奇迹的过程中，我和我的家人也失去了一些东西。我们离计算机越来越远了，而这种趋势在今天仍在继续：在用惯了顺手的iPad之后，我们甚至忘记了文件是怎样存储的。

不过，我还是给家里的苹果电脑安装了HyperCard。HyperCard是编程语言和开发环境的奇妙结合。你可以创建出虚拟卡，然后将它们拼接在一起，并添加表示特定功能的按钮和图标；你还可以制作出有趣的动画和酷炫的声效，然后把它们关联到其他卡上。或许你曾经玩过经典游戏《神秘岛》（Myst），它就是用HyperCard开发出来的。HyperCard就像一系列被保存在个人电脑上的原型网页，可以帮你做想做的任何事情。在当时，对于我这个刚开始探索计算机世界的孩子来说，这个可视化创作空间可谓是通向未来的完美窗口。

我用HyperCard创建了一个相当原始的密码生成器，它可以生成能用作密码的随机字符串，而且还带有一个可以让随机密码更易读写和长期记忆的选项。这个当然是比较简单的，但是在当时的我看来，它绝对领先于时代。

计算机游戏设计师哈伊姆·金戈尔德（Chaim Gingold）将HyperCard这类入门级软件工具称为“魔术蜡笔”。在童话《阿罗有支彩色铅笔》（Harold and the Purple Crayon）中，主人公哈罗德有支神奇的彩笔，用它画出什么，现实世界中就会出现什么。在金戈尔德看来，HyperCard就是这样的神奇工具，可以让“非程序员也能参与数字媒体的创作，并构建出动态的事物”。[16]人们普遍认为，苹果电脑已经“取消”了杂乱的代码，隐藏了计算机内部的运行状态，但即使真是如此，用户依然可以通过HyperCard造访复杂的数字领域。就我自己来说，HyperCard为我提供了一个可以窥探数字世界深处的舒适窗口，并为我提供了一些系统内部运行的线索。

能与我们产生交互的所有复杂系统都是有层级的。在技术系统中，我们通过深思熟虑的抽象化方法构建了这种分层结构；在自然界中，分层结构则建立在规模和进化的基础上。生物学中的层级从生化酶开始，到线粒体、细胞、器官，再到整个生命体，而后是整个生态系统。每个层级都可以提供不同层面的信息。当从一个层级的抽象化，向上扩展到另一个层级的抽象化时，我们或许会失去对细粒度（fine-grained）的理解和控制，但同时也能更好地理解更高级别的体系。在计算机软件中，我们的关注点可以从比特与字节转移到汇编语言，再到更高级别的计算机代码，最后到日常的用户界面，也就是我们点击、拖动的东西，例如对浏览器的使用。每一个更高的层级会都为我们带来更多的功能，但同时也会让我们更加远离底层逻辑。

当然，现在我们已经很清楚了，人类是不可能完全理解底层逻辑的。但是，我们至少应该窥探一下机罩底下的世界。如果将日常使用的平板电脑和手机视为会变魔术的玻璃和金属抛光板，完全不去理会面板下面或数字键下面到底发生了什么，那么我们肯定会失去一些机会。事实上，这确实是个隐患：如果所使用的系统是完全自动化的，[17]那么当它出现问题时，我们几乎无能为力。我们被屏蔽了，无法了解技术系统的内部运行，这种状态被称为“隐藏的电子复杂性”。换句话说，我们所用的设备存在着惊人的复杂性，但这种复杂性却被完全隐藏了起来。[18]

在20世纪60年代的电话通讯系统中存在着这样一个设计：当系统检测到故障时，就会将用户拨号关联到错误的号码上。[19]这个设计的目的是将技术系统故障定向为人为错误，以逃避人们对技术系统的苛责，用户会认为是自己拨错了号码。因为不了解内情，所以那些“拨错了号码的人”对电话通讯系统的权威性和神秘性有着自己的理解；一旦让他们知道了电话通讯系统是庞杂的、不完美的“人造物”，他们的理解就会变得不一样。

是的，我们必须打开机罩或通过窗口往里面瞥上一眼，看看那里发生了什么，尽管这种“管中窥豹”的做法通常无法做到“知全豹”。当技术的复杂性超出了人们的理解能力时，这样的“一瞥”，无论是对专家而言，还是对普通用户来说，都至关重要。仅仅让一个人，或一小部分人看到底层逻辑，并认识到系统和人类的局限性，是远远不够的。人人都应该关注，如何实现这“一瞥”。倘若做不到，我们在系统面前便会失去谦卑之心和敏锐直觉，从而转向崇敬或恐惧。关注技术底层，不仅有趣且有教育意义，而且还有助于抵制对待技术的“不正之风”。在当今这个纠缠时代中，我们越来越需要这种抵御能力了。

不过，如果很难瞥见机罩下的东西，我们又该怎么办呢？如果一个系统复杂到难以构建窗口，或是所提供的线索太少，我们又该如何是好？这里可以考虑另一种方法：构建模型。通过构建模型，我们可以推测出一个复杂的技术系统是如何运行的。

例如，虽然我们无法真的控制气象，也无法理解它所有的非线性细节，但却可以相当好地预测它、适应它，甚至为坏天气的到来做好准备。气象模型非常复杂，尽管它每个部分的设计都不是很难。我们不仅希望利用气象模型来指导日常活动，比如何时清理衣橱，何时出远门等，还希望对气象模型有所理解，哪怕是不完全的理解。当然，在面对突如其来的暴风雪或洪水时，我们还希望能靠它避险。

就像构建气象模型一样，我们也可以构建技术系统模型，哪怕有所简化也没关系。通过对感兴趣的系统进行模型构建，并测试其极限，调整其参数，我们便可以获得对复杂系统的强大洞察力，而用不着完全理解它。而且，构建模型本身也是一项值得培养的技能。

电脑游戏《模拟城市》就是这样的一个模型。[20]通过观察城市的运转模式，用户的洞察力得到了锻炼。在《模拟城市》出现之前，我一直不相信，没有城市规划或土木工程等专业背景的普通人，对城市的运转模式也能有清晰的理解。当然，我们无法扭转城市发展的进程，也无法

进行反事实的实验。在《模拟城市》出现之后，我们或许仍然无法完全理解现实中城市的复杂性，但是一般来说，玩过这个游戏的人对城市的运转模式确实有了更深刻的理解。我们不仅需要更好的与技术世界有关的模型或游戏，还要教会学生如何“玩”一些系统，测试它们的极限，探析它们的工作原理，至少在某种程度上去探析。这种“玩”，其实是模拟技术故障，并观察系统反应，可以让我们更舒适地与庞大且笨拙的系统相处，并更好地理解这个日益复杂的技术世界。

我们还需要一些类似于电视上的气象学家的，能够对系统内部事态作出说明的解释者。经济学家泰勒·考恩在其《平均时代的终结》一书的结尾部分曾预言，这种解释者即将登场。[21]在他看来，他们将会“磨炼自己寻找、吸收和评估这方面信息的技能……他们将会成为机器网络世界的真相阐释者……至少，在未来相当长的一段时期内，只有他们能够清楚地认识到这个世界所发生的事情”。这些解释者对模型构建的评价可能会相当高，因为那可以给我们带来一些关于复杂系统的灵感。[22]随着时间的推移，技术将变得越来越难以渗透，为此，中学和大学教育需要更多地引入模型构建和简化直觉，并把它们结合起来，作为必要的复杂性科学方法传授给学生，让每个人都有机会成为系统的解释者。

简化直觉是由复杂性科学家发展起来的一般模型，它为启发式方法奠定了基础。启发式方法可以使我们得以理解系统，同时又不会被其复杂性压倒，并能与“包容性的将就”相结合，这也是约翰·盖尔的“一般系统滑稽论”所采用的视角。这些用来窥探大规模复杂系统的小窗口，体现了我们对待技术的谦卑之心。

另外，在面对复杂的技术系统时，我们还可以采用另一种更加积极的思维方式。

## 以欣慰感看待不理解的事物

在意第绪语（Yiddish）中有一个非常有意思的词naches。这个词的含义是为他人感到骄傲或喜悦，也就是所谓的欣慰感。在意第绪语中，人们经常说，要学会shep naches，意为要学会从亲近之人，尤其是自己孩子的成就中捕获替代性的自豪感。这是最纯粹的愉悦感，也是你在酒吧、婚礼和毕业典礼上经常听人说起的一种快乐感受。

移民者一般都有一个执念：让下一代生活得比自己好，让第三代、第四代繁荣兴旺。对于父母来说，后代“必须”比自己更聪明、更成功。我们之所以会因下一代的成功倍感欣慰，原因之一便是我们受到了这个执念的驱使。那么，我们为什么不可以到技术成就中去捕获这种欣慰感呢？

我们或许不了解机器或系统正在做什么，在某些情况下，它们甚至可能是由多种技术构成的，但无论如何，它们都是人类智慧的“儿子”，是我们智力的“后代”，因而技术成就也可以令我们倍感欣慰。数千年以来，人类一直非常珍视这种情感，它给我们带来了巨大的快乐。现在，我们只需要将为人父母的自豪感转移到技术领域即可。

那么，这种欣慰感对于技术而言究竟意味着什么？最基本的，我们这些机器的创造者可以从技术“后代”的“成就”和“能力”中获得快乐。以计算机程序为例，除了其他成就之外，计算机程序现在已经能够创作出复杂的绘画作品和音乐作品了。加利福尼亚大学圣克鲁兹分校的作曲家戴维·科普（David Cope）所开发的一个软件能够按照给定作曲家的风格创作出新颖的音乐，而且这些音乐作品听起来非常不错。[23]比如，由这个软件创作的美国作曲家斯科特·乔普林（Scott Joplin）风格的作品，听起来就像是乔普林本人创作的一样。尽管科普并没有直接参与创作，但他仍然会为这个程序的“成就”感到自豪。他一手打造的计算机程序所创作的作品为他带来了很大的快乐。[24]同样，超级电脑沃森（Watson）的创造者、IBM的程序员们也会为沃森在国际象棋比赛中战胜了人类棋手而欣喜若狂。

我们可以将这种欣慰感延伸到更广阔的领域中。许多人都有自己支持的球队，并会为它的胜利而倍感自豪，尽管其实他们与球员们并无交集。当自己国家的运动员在奥运会上夺冠时，当自己国家的公民因重大发现而获得权威奖项时，我们都会感到兴奋无比。同样的道理，人类也可以为机器所取得的“成就”而感到非常欣慰，毕竟它们是由人类创造的，即使它们的成就不是我们的个人成就，即使我们无法完全理解它们。事实的确如此。从苹果手机到互联网，技术进步让许多人倍感欣慰，并心生感恩，尽管他们不一定了解这些技术的工作原理。

即使我们所构建的机器的复杂性超出了人们所能理解的范畴，我们也无须感到不安或失望。当我们的孩子创造出了令人惊讶的作品，而我们却无法理解时，我们是不会感到绝望或担心的；相反，我们会为他们的成功感到高兴，甚至会心生感恩。对于技术系统的“作品”，我们完全可以给予类似的回应。

“欣慰感”是一个框架，能够帮助我们认识到，我们正循着祖先数千年来的足迹前进着。[25]在前行的道路上，越来越少有人能够理解“人造世界”中最复杂的部分了。较之以往，最近的趋势也算不上新鲜或不同。

通过“欣慰感”这个镜头折射出来的谦卑之心，实为一种乐观主义精神，它让我们对自己所构建的这个“不可理喻”的世界心怀希望。除此之外，谦卑还能让我们在不断的追求中始终保持平衡：既能把握所构建事物的所有维度，又能看清自身的局限性。当然，对于专家来说，这是一种使命。

而且，谦卑还能帮助我们看清神秘与奇迹之间的区别。

## 谦卑之心 + 迭代的生物学思维

《奇迹世界》（The World of Wonders）是一本出版于19世纪下

半叶的书，书中汇集了许多新奇的想法、迷人的事物和离奇的事件。[26]从创作风格和写作手法看来，作者显然是一个热爱生活、享受生活，并充满好奇心的人，并不断地将人类进步与自然奇观交织在一起。这本书的内容涉及“历史和哲学中的一切奇思妙想、科学奇迹、眼镜商的玻璃，以及化学家通过努力所揭示的动物生命奇迹”等。当然，那些都是维多利亚时代的奇迹。这本书还展示了人类在周遭世界中寻找奇迹的多种方式。但是，奇迹世界并不是无法解释的世界，我们的每一次进步、每一项技术，都令人着迷并充满了力量。除此之外，最重要的是，它们皆能被理解，一点儿也不神秘。

对于复杂的技术系统中的神秘性，人们通常会有两种潜在的极端反应。第一种反应是有意且极力地削弱它的重要性，认为系统中根本不存在任何神秘的东西。对于系统的运行方式，这些人会编织出一些简单的故事。这些故事也许很吸引人，但都严重削弱了系统的复杂性。[27]他们认为，自己对正在使用的技术了如指掌，任何问题都是能解决的小问题。许多大型的科技公司都是这种风格，他们把所有意想不到的系统行为统统归结为小问题，还认为系统自身已经在处理这些问题了。

另一种极端反应则来自那些醉心于寻找神秘和未知事物的人。这些人并不在乎这种事物是否真的存在。他们通常都是技术的门外汉，总是把设备或系统的内部运行神秘化，并以此为荣；他们对苹果手机或电网的工作方式赞不绝口，并称其为魔术。这些人会问自己的父亲“电灯和真空吸尘器的工作原理是什么”，并满足于“简直就是魔术”之类的答案。不过，大多数人的反应都处于上述两种极端反应之间。他们可能会承认，在系统中存在一定程度的神秘性，但同时又希望自己可以理解这个世界。

当我们把奇迹和赞美，以及对神秘世界的感知都考虑进去时，事情就变得更加混乱了，我们甚至可能会“喜欢上”某种现象的复杂的美感。但是，从《奇迹世界》这本书中可以清楚地看到，奇迹不一定要以

牺牲可理解性为代价，反之亦然。事实上，许多人都发现，在面对一个非常庞大且复杂的系统时，自己理解得越透彻，应对得越自如，内心就会越愉悦。

例如，唐·诺曼曾经写道，如果能真正理解棒球比赛中的内野高飞球[28]规则，那将是一件多么令人开心的事情啊。棒球比赛的规则非常复杂，其规则手册多达200页，其中关于内野高飞球的规则还包含了大量的例外情况和补充说明：“在两人出局以前，当第一、二垒有跑垒员，或第一、二、三垒都有跑垒员时，击球员击出界内高飞球，且不含平飞球，也不含试图触击，同时内野手在正常守备条件下可以接住该球时，这个球即为内野高飞球。”棒球迷们花了很多的时间和心思去理解这个复杂的规则，有的人甚至觉得这里面隐藏着奇迹般的东西。

那么，我们在面对自己所构建的大型系统时，又该如何平衡其间的神秘和奇迹呢？特别是对于那些无论我们多么努力都无法完全理解的系统。

我们绝不能屈服于“将这两种感觉融合到一起”的想法，那会将神秘与奇迹混淆起来。[29]一些人认为，未能解释的东西定会激励我们继续前行。他们把神秘的东西错误地等同于神奇的东西，并据此认为，未能理解的东西也定是令人惊叹的。同样，我们也必须抵制“一旦被完全理解，就不再是奇迹”的想法。技术变革常常会引发这种思想。技术的发展日新月异，以致我们忘记了，我们对宇宙的理解以及理解宇宙的能力已经变得何其令人震惊；也忘记了其实直到最近我们才拥有了在屏幕上呈现三维图像，在地球上不间断通信，以及通过点击鼠标观看数十年前的电视广告的能力。不仅如此，那些让“不可能成为可能”的基础知识，也没能激发起人们心中的敬仰之意和惊叹之情。到最后，我们能感受到的只有“一个已经被看透、被控制的世界所留下的悲伤的惰性”。[30]这就是哲学家休伯特·德莱弗斯（Hubert Dreyfus）和肖恩·多兰斯·凯利（Sean Dorrance Kelly）所描述的那种风险：随着科

技力量的崛起，人们心目中的奇迹将出现得越来越少，以致一些人只能成天幻想“迷人的过去”。

我们必须努力维持这样两种相互对立的状态：神秘而无奇迹、神奇而不神秘。第一种状态要求我们努力克服自己的无知，[31]绝不能沉迷其中；第二种状态意味着，一旦理解了某个事物，我们就不会认为它是理所当然的了。

在我看来，从本质上说，上述两种状态的最终诉求都是建立科学的思维模式，而科学的思维模式是我们学习新事物和解决难题的必备能力和先决条件。如果我们对自己无意中创造出来的神秘事物抱有太多好奇的话，那么原本想要“消除神秘事物、厘清纠缠时代”的想法就会被削弱。

我们永远无法完全理解那些拼凑而成的复杂系统。我们始终会面对一些难解之谜，但是没关系，只要不害怕、不沉迷，就可以找到最好的方式——谦卑，哪怕携有一丝崇敬也无所谓，毕竟那是我们自己所构建的系统。谦卑、好奇，以及欣慰感，或许就是我们能做到的最好的状态了。我们必须坚持运用生物学思维，即使尚未实现对系统的充分理解。如果失败了，我们可以再谦卑一点，但绝不能气馁；我们定会找到让直觉深入技术系统底层，窥其堂奥的途径。

当丰田公司生产的汽车出现意外加速事故并导致死亡时，专家们无法准确地判断出故障原因，但他们给出了一个建议：通知相应型号丰田汽车的驾驶员，如果汽车出现自动加速并失去控制的情况，最快捷的解决方案就是立即将汽车挂到空挡上。[32]当然，作为一个消费者，你已经将家人和朋友，以及自己的生命“托付”给了一辆汽车，而这台强大的机器却可能以如此不可预知的、可怕的方式行事，那么恐怕你并不想听到这样一个建议。这就像是你摔断了骨头，庸医却建议你贴上一个创可贴。

但是，如果我们能明白，罕见的故障和无法解释的错误都是复杂性

的衍生物；如果我们能知道，即使是最警觉的设计师也无法预测全部故障和错误，那么在遇到这种状况时，我们就能从容不迫，而不会手足无措了。这也就是说，如果消费者能认识到，这种混乱的、难以理解的系统是现实新世界的一部分，[33]那么就不会再认为“立即换成空挡”是面对故障时最糟糕的选择了。[34]事实上，在当今时代，设计师和驾驶员已经成了探索纠缠时代的搭档，至少在以下这个非常重要的方面，他们并不像看起来那样毫无关联：谁都不可能完全预测复杂系统的所有行为。许多人都认识到了这一点，只是采用了不那么危险的处理方式，比如说，当计算机出现故障时，有些人会在沮丧中下意识地敲打敲打机身，或者重启计算机，希望系统能够自行解决问题。

进入20世纪后，我们迎来了无数的限制性定理（limitative theorems）。这些定理为我们划定了可知和可理解的知识边界。在数学领域，库尔特·哥德尔（Kurt Gödel）证明了，在某个给定的数学系统中，总是存在永远不能被证明为真或为假的陈述；在计算机科学领域，艾伦·图灵证明了，任何机器所能做到的事情都存在极限，无论某个特定的算法可能会发展到多么强大的程度。显然，这些领域并没有因此而消亡，甚至算不上遭遇了巨大挫折。尽管受到了限制，但它们在许多方向上都结出了累累硕果，其发展远远超出了那些为它们划定边界之人的想象。

在构建和使用复杂的技术系统时，我们试图了解它们如何运行以及如何失败，但是这种了解确实存在着极限。不过，这并不意味着我们要停止创造；恰恰相反，它只是意味着，当我们继续构建这类系统时，要明白它们还会不断成长，会变得更加奇异、更加复杂。因此，我们必须重新调整期望值。

从现在起，承认自己无法完全理解这些系统吧。这种做法将会改变我们理解和应对系统的方式。如果你参与了构建，那么在组装系统时，你就会意识到自己的理解能力是有限的；如果你与系统进行了交互，那

么你会认识到繁杂和意外是常态，而不是例外；如果你试图彻底消除这种常态，那么你不但会失败，还会让事情变得更糟。

科幻小说作家威廉·吉布森（William Gibson）经常提到“不可思议的现在”（unthinkable present）这个短语，这也是他笔下众多故事的背景。[35]无疑，这恰是我们所处的这个纠缠时代的内核。但是说到底，这些系统并不是完全无法被理解的，其性质和特征也不会永远不可言说。这个故事并没有以“对不可理解之物的恐惧”或“对不可思议之物的好奇”而结束。谦卑之心，再加上迭代的生物学思维，就是洞悉复杂世界的正确方式，会指导我们与复杂系统进行交互。但是，无论我们所持观点的细节如何，我们都无须屈服于路德维希·维特根斯坦的逻辑：“凡是不可言说之事，我们必须保持沉默。”

事实上，对于那些无法被完全理解、无法被完美处理的事物，我们还有很多话可以说！人类，将在这个纠缠时代继续生存下去，并发展壮大。

# 注释

## 引言　认识复杂系统

1.　有人猜测，《华尔街日报》的网站之所以会崩溃，是因为有太多用户为了阅读关于“纳斯达克停止交易”这则新闻而在同一时间段内登陆。请参阅：Jose Pagliery, “Tech Fail! Explaining Today's 3 Big Computer Errors,” CNN Money, July 8, 2015。关于此事件及其影响的进一步分析，请参阅：Zeynep Tufekci, “Why the Great Glitch of July 8th Should Scare You,” The Message, July 8, 2015。

2. Andrea Chang and Tracey Lien, “Outages at NYSE, United Airlines, WSJ. com Expose Digital Vulnerabilities,” July 8, 2015.

3. Thomas Homer-Dixon, The Ingenuity Gap: Facing the Economic, Environmental, and Other Challenges of an Increasingly Complex and Unpredictable Future (New York: Alfred A. Knopf, 2000; repr. Vintage, 2002), 171.

4. Edsger W. Dijkstra, “On the Cruelty of Really Teaching Computing Science,” E. W. Dijkstra Archive: The manuscripts of Edsger W. Dijkstra, 1930—2002, document no. EWD1036, December　1988．艾兹格·迪科斯彻在1972年受美国计算机协会（ACM）之邀举办了图灵讲座（Turing Lecture），并阐述了类似观点。请参阅：E. W. Dijkstra, “The Humble Programmer,” Communications of the ACM 15, no.10 (1972): 859－86。尽管不

是每一项技术都具有可计算性，但迪科斯彻的确深刻地指出了当今这个技术时代的生活模式。

5. Lee Billings, “Brave New Epoch,” Nautilus 009: January 30, 2014.

6. Neil Johnson et al., “Abrupt Rise of New Machine Ecology Beyond Human Response Time,” Scientific Reports 3: 2627, September 11, 2013.

7. 原文为：回到2008年，那是布拉德·胜山（Brad Katsuyama）第一次意识到股票市场已经变成了“黑箱”，普通人不可能理解它的内在工作机制。请参阅：Michael Lewis, “The Wolf Hunters of Wall Street,” The New York Times Magazine, March 31, 2014。

8. Berkeley J. Dietvorst et al., “Algorithm Aversion: People Erroneously Avoid Algorithms After Seeing Them Err,” Journal of Experimental Psychology: General 144, no.1(2015), 114－26.

9. 关于这种崇敬心理的实例有：一位记者写下了自己在参观谷歌数据中心时的感受；请参阅：Stephen Levy, “Google Throws Open Doors to Its Top-Secret Data Center,” Wired, October 17, 2012。另一位记者则描述了自己对Facebook新闻推送算法的崇敬之心，他是这样说的：“新闻推送算法在评论界掀起了轩然大波，它似乎有了自己的思想，就像是有智慧的北欧古符文被解除了封印，开始追逐一些超出人类理解能力的目标。”请参阅：Will Oremus, “Who Controls Your Facebook Feed,” Slate, January 3, 2016。

10. 关于“对算法的崇拜”的进一步讨论，请参阅：Ian Bogost, “The Cathedral of Computation,” The Atlantic, January 15, 2015。

# 01　欢迎来到这个纠缠的时代

1. James Gleick, “Richard Feynman Dead at 69; Leading Theoretical Physicist,” The New York Times, February 17.

2. 关于丰田汽车意外加速事故的更多信息，请参阅：Ken Bensinger and Jerry Hirsch, “Jury Hits Toyota with $3-million Verdict in Sudden Acceleration Death Case,” Los Angeles Times, October 24, 2013; Ralph Vartabedian and Ken Bensinger, “ Runaway Toyota Cases Ignored,” Los Angeles Times, November 8, 2009; Margaret Cronin Fisk, “Toyota Settles Oklahoma Acceleration Case After Verdict,” Bloomberg Business, October 25, 2013; Associated Press, “Jury Finds Toyota Liable in Fatal Wreck in Oklahoma,” New York Times, October 25, 2013。

3. 2014年10月3日，库普曼在他的博客上发表了一个题为“丰田汽车意外加速事故与软件安全案例研究”的演讲。也请参阅：the report by Michael Barr of the Barr Group on Bookout v. Toyota。

4. 在内在固有复杂性与偶发复杂性之间，存在着一条界线。前者是系统正常运行所需的复杂性，包括为例外情况和特殊情况提供预案的各种机制；后者则更接近于本书所说的“过度复杂”，往往出现在系统不按计划超速“生长”的时候。

5. 关于阿丽亚娜5型火箭的事故详情，请参阅：Homer-Dixon, The Ingenuity Gap。霍默－狄克逊的部分叙事基于James Gleick, “A Bug and a Crash: Sometimes a Bug Is More Than a Nuisance,” 1996(which originally appeared in The New York Times Magazine, December 1996)。关于这类系统突然崩溃的直接原因与深层原因的讨论，请参阅：Chris Clearfield and James Owen Weatherall, “Why the Flash Crash Really Matters,”

Nautilus 023, April 23, 2015。

6. Clearfield and Weatherall, “Why the Flash Crash Really Matters.”

7. 从根本上说，这类事故皆可归因于内在复杂性，也就是存在于大型系统内部的，随时间推移而不断进化的复杂性，而不能归因于某个特定的外生冲击。

8. 我们必须认识到，“挑战者号”航天飞机的失事原因比坊间所传要复杂得多。例如，有种说法是，在发射之前，某些工程师已经预感到这架航天飞机可能会发生“灾难性事故”；尽管他们无法说出准确原因，但是他们的确反对过这次发射。当时，有工程师指出“温度是一个因果性的因素”，但并不确定。请参阅：Wade Robison et al., “Representation and Misrepresentation: Tufte and the Morton Thiokol Engineers on the Challenger,” Science and Engineering Ethics 8, no.1(2002): 59–81。在这个事故的“口述”材料中也有一些线索可以表明，某些工程师似乎知道失事原因；理查德·费曼也得到了这方面的材料，并在听证会上做出了强调。请参阅：Margaret Lazarus Dean, “An Oral History of the Space Shuttle Challenger Disaster,” Popular Mechanics, February, 2016。

9. Philip Ball, “Science Fictions,” Aeon, October 29, 2012.

10. Ian Beacock, “Humanist among Machines,” Aeon, June 25, 2015.

11. Max Weber, “Science as a Vocation,” in From Max Weber: Essays in Sociology, trans. and ed. H. H. Gerth and C. Wright Mills, 129–56(New York: Oxford University Press, 1946, repr.1958) .

12. 人类一手创造之物，令人类困惑不解。类似的观点，法国诗人保罗

·瓦莱里（Paul Valéry）早已“预言”：“因此，所有的问题归根结底是在问：人类心智能不能掌握人类心智的创造物？”转引自：Langdon Winner, Autonomous Technology: Technics-out-of-Control as a Theme in Political Thought (Cambridge, MA: The MIT Press, 1977), 13。

13. 这只是两个术语之间可能存在的若干区别之一。

14. 感谢亚伦·克劳塞特。在我们对本书进行第一次讨论的时候，他为我提供了这个水上浮标的例子。

15. Kevin Kelly, “Cities are technological artifacts, the largest technology we make.” What Technology Wants (New York: Viking, 2010), 81.

16. David McCandless, “Codebases: Millions of Lines of Code,” infographic, v.0.9, Information Is Beautiful, September 24, 2015. 假设一本百科全书有3万页，每一页可以放下1 000行代码，那么据推算，苹果操作系统需要好几本百科全书才能登载完。

17. 有信号灯的交叉路口数量是根据美国交通部联邦公路管理局2015年10月20日的数据做出的估计。

18. Gideon Lewis-Kraus, “The Fasinatng...Frustrating...Fascinating History of Autocorrect,” Wired, July 22, 2014.

19. Laura Saunders, “Paper Trail,” sidebar to article “Don't Make These Tax Mistakes: Fifteen Common Tax-Filing Errors That Can Cost You Dearly,” The Wall Street Journal, January 31, 2014.17.

20. 对复杂系统不同层级的理解，请参阅：Stephen Jay Kline, Conceptual Foundations for Multidisciplinary Thinking

(Stanford, CA: Stanford University Press, 1995), 268。

21. Erik Sandberg-Diment, “A Computer Comes in from the Cold,” The New York Times, April 21, 1987.

22. Gerard J. Holzmann, “Code Inflation,” IEEE Software (March/ April 2015): 10 – 13.

23. Philip K. Howard, “Fixing Broken Government: Put Humans in Charge,” The Atlantic, September 22, 2014.

24. Quinn Norton, “Everything is Broken,” The Message, May 20, 2014.

25. Winner, Autonomous Technology, 290 – 91.

26. Danny Hillis, “The Age of Digital Entanglement,” Scientific American, September 2010, 93.

27. Nick Bostrom, Superintelligence: Paths, Dangers, Strategies (Oxford, UK: Oxf ord University Press, 2014), 17．不过，此次闪电崩盘的原因至今不明。

28. César Hidalgo, Why Information Grows: The Evolution of Order, from Atomsto Economies (New York: Basic Books, 2015).

## 02 复杂系统形成的4个原因

1. Barry M. Leiner et al., “Brief History of the Internet,” Internet Society, October 15, 2012.

2. Randall Munrose, “DNA,” xkcd, November 18, 2015.

3. Chris Kirk, “Battling My Daemons: My Email Made Me

Miserable. So I Decided to Build My Own Email App from Scratch," Slate, February 25, 2015.

4. 计算机代码与法律代码、道德代码等其他代码所具有的相似性，还有很多可以讨论的，但是最好将它们视为不同的系统代码。

5. 请参阅：《美利坚合众国宪法》，第一条，第八款："To Establish Post Offices and Post Roads"；"Title 39-Postal Service," Code of Federal Regulations (annual edition),revised July 1, 2003。

6. Michael J. Bommarito II and Daniel M. Katz, "A Mathematical Approach to the Study of the United States Code," Physica A: Statistical Mechanics and its Applications 389, no.19(2010): 4195–200.

7. "Wright 1903 Flyer," NASA Glenn Research Center.

8. "747 Fun Facts," Boeing Commercial Airplanes.

9. Kelly, What Technology Wants, 279.

10. Kelly, What Technology Wants, 279; David McCandless, "Codebases: Millions of Lines of Code," infographic, v.0.9, Information Is Beautiful, September 24, 2015.

11. McCandless, "Codebases."

12. M. D. Fagen, ed., A History of Engineering and Science in the Bell System: The Early Years(1875—1925), Technical Publication Department, Bell Laboratories, 1975.

13. 美国联邦航空管理局担心"千年虫"问题的说法源自：Homer-Dixon, The Ingenuity Gap。

14. Matthew L. Wald, "Warning Issued on Air Traffic

Computers,” The New York Times, January 13, 1998．按照霍默－狄克逊的说法，他们雇用了一位退休程序员去帮助解决这个问题。

15. “IRS Trudges On with Aging Computers,” CNET News, April 12, 2007；John Bodoh, “Tech Timebomb: The IRS Living in the 1960s,” Washington Examiner, December 17, 2014.

16. “The Shuttle: NASA's IT Legacy,” Information Age, July 18, 2011.

17. Frederick P. Brooks Jr., The Mythical Man-Month: Essays on Software Engineering, anniversary ed.(Boston, MA: Addison-Wesley, 1995; orig. pub.1975), 53．引用的这句话似乎是一句拉丁谚语，被误传为出自古罗马诗人奥维德（Ovid）之口。

18. 这个术语，霍默－狄克逊在《创造力差距》一书中也有使用。

19. Stewart Brand, The Clock of the Long Now: Time and Responsibility (New York: Basic Books, 1999), 85.

20. 这种说法源自斯图尔特布兰德的一篇文章。他在文中写道：“除了数据格式和数字存储媒介已经消失之外，还存在一个更深层级的问题。大型计算机系统是推动企业、公共机构，以及金融部门发展的核心。随着时间的推移，这些庞大的系统变得异常复杂和不可理解。这是因为新的特性被不断加入，旧的问题通过一层又一层的补丁得到了处理；除此之外，一代又一代的程序员还加入了新的编程工具和风格，系统的某些部分也被重新设计并赋予了新的功能。出于尊敬，也出于厌恶，计算机专业人士称这些‘怪物’为‘古迹’。从这些被称为‘古迹’的系统中引申出新的功能，绝不是一声令下就能完成的事情；只能小心翼翼地进行炼金术般的实验，如果运气好的话，最终也可能会得到想要的结果。”请参阅：“Written on the Wind,” The Long Now Foundation, February 11, 1998。

21. Christopher Ingraham, “Charted: The Skyrocketing Complexity of the Federal Tax Code,” Washington Post, April 15, 2015.

22. Cheek v. United States, 498 U. S 192.; United States v. Murdock, 290 U. S.389.

23. Susan E. Dudley and Jerry Brito, Regulation: A Primer, 2nd ed. (Arlington, VA, and Washington, DC: Mercatus Center, George Mason University, and The George Washington University Regulatory Studies Center, 2012), 5.

24. The Economist, November 19, 1955. 不过，当一个国家处于战争状态时，这种关系似乎并不成立。

25. Meir M. Lehman, “Programs, Life Cycles, and Laws of Software Evolution,” Proceedings of the IEEE 68, no.9(1980): 1060-76.Also Lehman et al., “Metrics and Laws of Software Evolution-The Nineties View,” in METRICS' 97, Proceedings of the Fourth International Software Metrics Symposium (Washington, DC: IEEE Computer Society, 1997). 另外也请参阅下文中要讨论的约翰·盖尔所著的《系统圣经》。

26. 这种重写其实相当于“重构”，“重构”即重新编写软件代码，使软件的内在更统一、更“干净”，但不一定会改变其外部功能。

27. 宇航员艾伦·谢泼德（Alan Shepard）曾经说过：“虽然身处太空，心里却很清楚，你的安全完全取决于政府招标时出价最低的那个竞标者。一想到这里，不禁心中酸楚。”

28. John Kelly, “Look Out Below: Danger Lurks Underground from Aging Gas Pipes,” USA Today, September 23, 2014.

29. Metropolitan Transportation Authority (MTA), “CBTC:

Communications-Based Train Control," July 20, 2015.

30. 而且，为了预测边界情况和例外情况，这些系统甚至不得不在复杂状态下启动，本章下文将对此进行讨论。

31. Harry McCracken, "Fifty Years of BASIC, the Programming Language That Made Computers Personal," TIME, April 29, 2014. 对于这类分叉和循环，更严格的控制方法是运用更明确的指令语句，比如for循环语句。

32. Edsger W. Dijkstra, "GoTo Statement Considered Harmful," Communications of the ACM 11, no.3(1968): 147-48. 需要注意的是，迪科斯彻这篇论文最初的标题是"A Case Against the GOTO Statement"。

33. 其中一种方法是重构。软件开发者布莱恩·福特（Brian Foote）和约瑟夫·约德（Joseph Yoder）曾经阐释过，为什么"当前的标准软件构架"其实是"一个松散的，甚至有害的结构体系"，或者说是一个"大泥球"；他们还试图描绘出改进此类系统的内部通道。请参阅：Brian Foote and Joseph Yoder, "Big Ball of Mud," Fourth Conference on Pattern Languages of Programs, Monticello, IL, September 1997; in Pattern Languages of Program Design 4, ed. Brian Foote, Neil Harrison, and Hans Rohnert, chapter 29(Boston: Addison-Wesley, 2000)。

34. Koopman, "Case Study of Toyota Unintended Acceleration," slide 38. 库普曼指出，意大利面条式的代码可能是导致高度复杂性的原因之一。总部位于英国的汽车工业软件可靠性联合协会（MISRA）发布了行业标准《MISRA软件指南》，试图提高系统的安全性能。

35. Philip K. Howard, The Rule of Nobody: Saving America from

Dead Laws and Broken Government (New York: W. W. Norton, 2014).

36. Howard, The Rule of Nobody, 8.

37. Howard, The Rule of Nobody, 12。以新桥取代旧桥的建设项目通常需要10年时间才能获批立项；至于高速公路建设项目，单是环境评估环节就要花上8年。

38. Michael Mandel and Diana G. Carew, Regulatory Improvement Commission: A Politically-Viable Approach to U. S. Regulatory Reform, Progressive Policy Institute policy memo (Washington, DC: Progressive Policy Institute, May 2013).

39. John Palfrey and Urs Gasser, Interop: The Promise and Perils of Highly Interconnected Systems (New York: Basic Books, 2012).

40. Sergey V. Buldyrev et al., "Catastrophic Cascade of Failures in Interdependent Networks," Nature 464(2010): 1025–28; Natalie Wolchover, "Treading Softly in a Connected World," Quanta Magazine, March 18, 2013.

41. 失败的成本实际上是一个假想的分布，而不是某个特定的数字；不过，我们也可以将其想象为一个期望值或平均值。

42. J. R. Minkel, "The 2003 Northeast Blackout-Five Years Later," Scientific American, August 13, 2008.

43. 关于构建成本和故障成本的观点，要感谢爱德华·荣格（Edward Jung）在2014年3月8日的私人信件中给我的启发。

44. "A Maps App with Problems," The New York Times, September 27, 2012.

45. Jeronimo Cello et al., “Chemical Synthesis of Poliovirus cDNA: Generation of Infectious Virus in the Absence of Natural Template,” Science 297(2002): 1016-18; Eckard Wimmer, “The Test-Tube Synthesis of a Chemical Called Poliovirus: The Simple Synthesis of a Virus Has Far-Reaching Societal Implications,” EMBO Reports 7, no.1S(2006): S3 – S9.

46. Doug Hill, Not So Fast: Thinking Twice about Technology (Cellarius Press, 2013, out of print;Athens, GA: University of Georgia Press, forthcoming);Kevin Kelly, What Technology Wants.

47. Dijkstra, “On the Cruelty of Really Teaching Computing Science,” E. W. Dijkstra Archive: The manuscripts of Edsger W. Dijkstra, 1930-2002, document no. EWD1036, December 1988.

48. 系统的复杂性反映了世界的复杂性，请参阅：Vikram Chandra, Geek Sublime: The Beauty of Code, the Code of Beauty (Minneapolis: Graywolf Press, 2014)。我们之所以会编写出复杂的代码，并不只是因为这个世界太过复杂，但大多数时候的确如此。幸运的是，有一些方法可以缓解这个问题，请参阅：Steve McConnell, Code Complete: A Practical Handbook of Software Construction, 2nd ed.(Redmond, WA: Microsoft Press, 2004), 583。

49. 关于自动驾驶汽车的复杂性，请参阅：Astro Teller, “How to Make Moonshots,” Backchannel, March 17, 2015。

50. 一种解决方案是进行人工干预，即手动排除故障，至少要对例外情况进行“硬编码”。谷歌公司正是这样处理谷歌地图的：“这是一种实现超级可靠性的谷歌式（Google-y）方法。谷歌公司旗下许多有名的

驱动型计算项目，例如创建地图等，背后都隐藏着数以千计的员工。他们随时监督着系统，并修正系统的错误。这是谷歌公司公开的秘密之一。他们成功地让人工智能变身为催化剂。他们并没有使用编程的方式来构建最后这一道防线，构建这最后的1%或0.1%，甚至0.01%的可靠性；而是通过一部分廉价的人力来实现这一点。未来，人类可以通过指导机器运行，让自己从这项工作中脱身而出。‘如果人类告诉机器，这件事这样做是正确的，那么这个指令就会成为一个可以融入系统的信息；未来需要人工干预的情况就会逐渐减少。’消息人士这样说。”请参阅：Alexis C. Madrigal, “Inside Google's Secret Drone-Delivery Program,” The Atlantic, August 28, 2014。

51. “Friar Daw's Reply,” from Six Ecclesiastical Satires, ed. James M. Dean, TEAMS Middle English Texts Series (Kalamazoo, MI: Medieval Institute Publications, 1991)可以从罗切斯特大学罗宾斯图书馆数字项目网站（Robbins Library Digital Projects）上下载。

52. 请注意，包括长尾分布在内，不是所有的重尾分布都是服从幂律的。

53. András Kornai, Mathematical Linguistics (London: Springer-Verlag, 2008), 71．研究表明，在多种语料库中，罕见语所占比例约为40%到60%。

54. William A. Kretzschmar Jr., The Linguistics of Speech (Cambridge, UK: Cambridge University Press, 2009)．同样相关的，还有弗斯语言学（Firthian linguistics），它“基于这样一种观点：语言模式无法用单一的原则来分析，也无法用某个系统类型来解释”，而必须同时考虑多个与上下文有关联的系统。请参阅：David Crystal, ed., Dictionary of Linguistics and Phonetics, 6th ed. (Malden, MA: Wiley-Blackwell, 2008), 181。

55. Peter Norvig, “On Chomsky and the Two Cultures of Statistical Learning”.

56. 这个关于机器翻译的预言般的故事还有好几个版本。

57. Nick Bostrom, Superintelligence: Paths, Dangers, Strategies (Oxford, UK: Oxf ord University Press, 2014), 15.

58. 这个数字是我自己总结出来的，如果哪位语言学家有不同意见，请告知我。

59. Alon Halevy et al., “The Unreasonable Effectiveness of Data,” IEEE Intelligent Systems 24, no.2(2009): 8-12. 在某种意义上，和那些乍看起来规则既简单又优雅的模型相比，这些统计学模型其实更加简单，因为前者最后都会因各种始料不及的原因而变得复杂。

60. Douglas Heaven, “Higher State of Mind,” New Scientist 219(August 10, 2013), 32-35.

61. Brooks, Mythical Man-Month, 183-84. 小弗雷德里克·布鲁克斯认识到，复杂性的类型多种多样，甚至包括与软件相互作用的环境所施加的复杂性。

62. David G. Post and Michael B. Eisen, “How Long Is the Coastline of the Law? Thoughts on the Fractal Nature of Legal Systems,” Journal of Legal Studies 29, no.2 J(2000): 545-84.

63. Jack M. Balkin, “The Crystalline Structure of Legal Thought,” Rutgers Law Review 39, no.1(1986): 1-108.

64. Post and Eisen, “How Long Is the Coastline of the Law?”

65. 波斯特和艾森发现了幂律的基本特性。

66. Mark D. Flood and Oliver Goodenough, “Contract as

Automaton: The Computational Representation of Financial Agreements," OFR(Office of Financial Research) Working Paper no.15-04, March 26, 2015.

67. Duncan Watts, "Too Complex to Exist," The Boston Globe, June 14, 2009.

68. Palfrey and Gasser, Interop.

69. 假设每个独立模块能够以3种不同的方式"出站"连接到另一个模块，并以3种不同的方式"入站"连接到其他模块，或者自环连接，那么，我们将会得到大约$10^{32}$个潜在的网络。在当前人类可观测的宇宙范围中，恒星的准确数量尚不清楚，但据估计，应该不到$10^{30}$颗。请参阅：Elizabeth Howell, "How Many Stars Are In The Universe?" Space. com, May 31, 2014。

70. 本书的主要论题是如何在一个人类无法理解的时代生存下去，而今，这已经成为一个很现实的问题；尽管如此，我们仍然可以找到一些方法来设计和构建更具可控性的系统，比如"系统思维"这样的方法。请参阅：Nancy G. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety (Cambridge, MA: The MIT Press, 2011)。

71. McConnell, Code Complete, 521.

## 03 为什么复杂系统越来越难以理解了

1. 这个故事和分析取材于：Nancy G. Leveson and Clark S. Turner, "An Investigation of the Therac-25 Accidents," Computer 26, no.7(1993), 18-41。

2. Leveson and Turner, "An Investigation."

3. 机器，更准确地说，编程语言也不是不能从“1”开始计数，只不过，如今的大多数编程语言都是从“0”开始计数的。原因非常古老，就连很多程序员都不甚清楚，关于这段历史的记录，请参阅：Michael Hoye, “Citation Needed,” blarg? Mike Hoye's weblog, October 22, 2013。

4. Scott Rosenberg, Dreaming in Code: Two Dozen Programmers, Three Years, 4, 732 Bugs, and One Quest for Transcendent Software (New York: Three Rivers Press, 2008), 6-7.

5. Homer-Dixon, The Ingenuity Gap, 186.

6. 实际上是这样算的：10 000个名词×1 000个动词×9 999个名词，那么需要3万多年才能说完这些句子。

7. Steven Pinker, The Language Instinct: How the Mind Creates Language (New York: William Morrow, 1994; repr. Harper Perennial, 1995), 205.

8. Ray Kurzweil, The Age of Spiritual Machines: When Computers Exceed Human Intelligence (New York: Penguin, 1999), 95.

9. Mark Pilgrim, Dive into Python: Python from Novice to Pro, updated 2004．虽然“康德发生器”可以将从句多重嵌套进其他句子中，但这个程序似乎不能实现无限递归，因为它总会自动结束。

10. “Garden Path Sentence,” Wikipedia.

11. 还有一个例子是，我们进行战略性决策时所能考虑到的“步数”。如博弈论所述，当决策依赖于我们所认为的，其他人的想法和行为时，我们就会遇到这个问题。很少有人能够想到很多步，因为大多数人总是根据“他认为我认为他认为我认为……”的原则在做事。请参阅：

Colin Camerer et al., “Behavioural Game Theory: Thinking, Learning and Teaching,” Caltech Working Paper。

12. George A. Miller, “The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information,” The Psychological Review 63(1956): 81－97.

13. Lin Zhong, “Limitations of Human Mind,” lecture notes for ELEC513/COMP513, Complexity in Modern Systems, Department of Electrical and Computer Engineering, Rice University.

14. Bostrom, Superintelligence, 59-60．对于长期记忆，有一些人给出了很大容量的估值，比如2.5PB。

15. Thomas Hills and Ralph Hertwig, “Why Aren't We Smarter Already: Evolutionary Trade-Offs and Cognitive Enhancements,” Current Directions in Psychological Science 20, no.6(2011): 373-77.

16. Jorge Luis Borges, “Funes, His Memory,” in Collected Fictions, trans. Andrew Hurley(New York: Viking Penguin, 1998), 131-37.

17. Robert Kanigel, The Man Who Knew Infinity: A Life of the Genius Ramanujan (New York: Charles Scribner's Sons, 1991; repr. Washington Square Press, 1992).

18. Chandra, Geek Sublime, 48.

19. Jody Rosen, “The Knowledge, London's Legendary Taxi-Driver Test, Puts Up a Fight in the Age of GPS,” T, The New York Times Style Magazine, November 10, 2014.

20. 塞萨尔·伊达尔戈（César Hidalgo）创造了个人字节(personbyte)这个术语，用来描述人脑能够容纳的信息量和知识量，请参阅：Hidalgo, Why Information Grows。

21. John Symons and Jack Horner, “Software Intensive Science,” Philosophy and Technology 27, no.3(2014): 461－77.

22. 莱姆在几十年前就分析过这种不可理解的复杂性，请参阅：Lem, Summa Technologiae, trans. Joanna Zylinska(orig. pub. in Polish, 1964; Minneapolis: Unive rsity of Minnesota Press, 2013), 96-97。

23. Jim Gao, “Machine Learning Applications for Data Center Optimization,” Google White Paper.

24. Joe Kava, “Better Data Centers through Machine Learning,” Google Official Blog, May 28, 2014.

25. Douglas Heaven, “Higher State of Mind,” New Scientist 219(August 10, 2013), 32-35.

26. 需要注意的是，这种电路实际上是在硬件领域中发展起来的。“种群”中的每个成员都在一个被称为“现场可编程门阵列”（Field-Brogrammable Gate Array）的可编程电路中实例化，而不仅仅只是进行模拟运算。其他许多进化算法则完全发生在软件内部。请参阅：Adrian Thompson, “Exploring Beyond the Scope of Human Design: Automatic Generation of FPGA Configurations Through Artificial Evolution(Extended Abstract),” 8th Annual Advanced PLD and FPGA Conference, 1998。也可参阅：Thompson, “An Evolved Circuit, Intrinsic in Silicon, Entwined with Physics,” in Evolvable Systems: From Biology to Hardware, ed. Tetsuya Higuchi et al., Lecture Notes in Computer Science vol.1259(New

York: Springer, 2008), 390-405。

27. Kevin Kelly, Out of Control (New York: Basic Books, 1994), 338.

28. Steven Rosenbush and Laura Stevens, “At UPS, the Algorithm Is the Driver,” The Wall Street Journal, February 16, 2015. 在博客文章《边际革命》（Marginal Revolution）中，阿历克斯·塔巴洛克（Alex Tabarrok）把这种智能称为“不透明智能”（opaque intelligence）。

29. Tyler Cowen, Average Is Over: Powering America beyond the Age of the Great Stagnation (New York: Dutton, 2013), 72.

30. Feng-Hsiung Hsu, Behind Deep Blue: Building the Computer That Defeated the World Chess Champion (Princeton, NJ: Princeton University Press, 2002).

31. Flood and Goodenough, “Contract as Automaton.”

32. Donald Sull and Kathleen M. Eisenhardt, Simple Rules: How to Thrive in a Complex World (New York: Houghton Mifflin Harcourt, 2015), 12-13.

33. 这三本书分别是：Paula Findlen, ed., Athanasius Kircher: The Last Man Who Knew Everything (New York: Routledge, 2004);Andrew Robinson, The Last Man Who Knew Everything: Thomas Young, the Anonymous Polymath Who Proved Newton Wrong, Explained How We See, Cured the Sick and Deciphered the Rosetta Stone (New York: Plume, 2007; orig. pub. Saddle River, NJ: Pi Press/Pearson Education, 2006);Leonard Warren, Joseph Leidy: The Last Man Who Knew Everything (New Haven: Yale University Press, 1998)。

34. Philip Ball, Curiosity: How Science Became Interested in Everything (Chicago: University of Chicago Press, 2013), 55.

35. Ball, Curiosity, 156.

36. Daniel J. Boorstin, The Discoverers: A History of Man's Search to Know His World and Himself (New York: Random House, 1983), 414.

37. Boorstin, The Discoverers, 414.

38. Ball, Curiosity, 120.

39. Ball, Curiosity, 120.

40. John M. Ziman, Knowing Everything About Nothing: Specialization and Change in Research Careers (Cambridge, UK: Cambridge University Press, 1987),v.

41. Benjamin F. Jones, "The Burden of Knowledge and the 'Death of the Renaissance Man': Is Innovation Getting Harder?" The Review of Economic Studies 76(2009): 283-317．这个理论主要关注的是与技术进步有关的知识。

42. Benjamin F. Jones, E. J. Reedy, and Bruce A. Weinberg, "Age and Scientific Genius," in The Wiley Handbook of Genius, ed. Dean Keith Simonton(Malden, MA: John Wiley and Sons, 2014).

43. "Fascinating Facts," Library of Congress.

44. Edward O. Wilson, Consilience: The Unity of Knowledge (New York: Alfred A. Knopf, 1998; repr. Vintage Books, 1999), 42 – 43.

45. Jordan Bell-Masterson, "Innovation Series: The Rising

Costs of Invention,” Growthology, March 24, 2015.

46. Michael Ogawa and Kwan-Liu Ma, “Software Evolution Storylines,” SOFTVIS' 10: Proceedings of the 5th International Symposium on Software Visualization (New York: ACM Digital Library, 2010), 35-42.

47. Brooks, Mythical Man-Month, 180.

## 04 令人费解的bug

1. “但是，这几十年来，在我的脑海中始终有一些问题挥之不去。这个作弊法是故意被添加到代码中，让内线玩家作为后门来使用的吗？或者只是软件中的一个小故障，是连续几个版本的游戏只读存储卡中一直存在的某个意外问题所引发的副作用？如果这个作弊法只是一个小故障，那么代码到底出了什么问题？是否有其他动作序列可以更快地到达禁用状态？”请参阅：Chris Cantrell, “Arcade / Galaga,” Computer Archeology。我还了解到，不少人认为这是一个有意为之的骗局，例如下面这篇文章，就持有这种观点：Jason Eckert, “The Galaga No Fire Cheat Mystery,” October 31, 2012(updated May 2014)。我第一次知道《小蜜蜂》有这个问题，有赖于这篇文章：Clive Thompson, Smarter Than You Think: How Technology Is Changing Our Minds for the Better (New York: Penguin, 2013)。

2. 谷歌公司杰出的研究员乌尔斯·霍尔泽(Urs Hölzle)指出：“对于致力于推进宏伟计划的人来说，复杂性在某种意义上是邪恶的，因为它会让某些bug两三年才冒出来一次；而当你看到这些bug时，就意味着已经出了大事，因为它具有巨大的级联效应。”请参阅：Jack Clark, “Google: ‘At Scale, Everything Breaks,’” ZDNet, June 22,

2011。

3. A. M. Turing, “Computing Machinery and Intelligence,” Mind 59(1950): 433-60.

4. 在3万行代码中，每千行就会产生大约1万个特定类型的错误。请参阅：John Symons and Jack Horner, “Software Intensive Science,” Philosophy and Technology 27, no.3(2014): 461-77; J. K. Horner, “Persistence of Plummer-Distributed Small Globular Clusters as a Function of Primordial-Binary Population Size,” Proceedings of the International Conference on Scientific Computing(CSC) (Athens: CSREA Press, 2013), 100-106。

5. 有人估计，每3至5行代码就存在一个错误，这个数字无疑是令人吃惊的。请参阅：Roger A. Grimes, “In His Own Words: Confessions of a Cyber Warrior,” InfoWorld, July 9, 2013。

6. Bruce Brown, Nigel R. Smith, and Bruce Kratofil, The Windows 95 Bug Collection (Reading, MA: Addison-Wesley, 1996), 3-10.

7. McConnell, Code Complete, 652.

8. J. M. Carlson and John Doyle, “Complexity and Robustness.” PNAS 99, Suppl.1(2002): 2538-45.

9. Carlson and Doyle, “Complexity and Robustness.”

10. Koopman, “Case Study of Toyota Unintended Acceleration,” slide 20.

11. 这些著作包括：The Words： Anatomy of a City (2005)；The Heights: Anatomy of a Skyscraper (2011); The Way to Go:

Moving by Sea, Land, and Air (2015)。这几本著作均由企鹅出版集团出版。

12. “Massachusetts Water Crisis,” The Boston Globe.

13. Andrew Blum, Tubes: A Journey to the Center of the Internet (New York: Ecco, 2012).

14. Roger A. Grimes, “Shellshock Proves Open Source's ‘Many Eyes’ Can't See Straight,” InfoWorld, September 30, 2014.

15. Jon Brodkin, “Why Gmail Went Down: Google Misconfigured Load Balancing Servers(Updated),” Ars Technica, December 11, 2012.

16. Joshua Bloch, “Extra, Extra-Read All About It: Nearly All Binary Searches and Mergesorts Are Broken,” Google Research Blog, June 2, 2006．在以下论著中也有相关讨论：Chandra, Geek Sublime, 124。

17. 一个小故障是“浏览软件内部结构的可能性”，请参阅：Olga Goriunova and Alexei Shulgin, “Glitch,” in Software Studies: A Lexicon, ed. Matthew Fuller, 110-19 (Cambridge, MA: The MIT Press, 2008), 114。错误和bug可以为改进软件提供机会，可参阅纳西姆·尼古拉斯·塔勒布（Nassim Nicholas Taleb）的论著：Antifragile: Things That Gain from Disorder (New York: Random House, 2012)。关于塔勒布的“反脆弱”思想在软件开发中的应用，请参阅：Martin Monperrus, “Principles of Antifragile Software”。

18. 在很多地方都可以看到与温哥华证券交易所指数事件有关的故事。请参阅：Anne Greenbaum and Timothy P. Chartier, Numerical

Methods: Design, Analysis, and Computer Implementation of Algorithms (Princeton, NJ: Princeton University Press, 2012), 117; Kevin Quinn, “Ever Had Problems Rounding Off Figures? This Stock Exchange Has,” Wall Street Journal, November 8, 1983。

19. “混沌猴”可以在网上下载。

20. Bill Bryson, A Short History of Nearly Everything (New York: Broadway Books, 2003), 162.

21. George Dyson, Turing's Cathedral: The Origins of the Digital Universe (New York: Pantheon, 2012), 4.

## 05 为什么需要生物学思维

1. Dictionary of National Biography, 1885–1900, vol.18, “Fairfax, Nathaniel, M. D.”需要注意的是，这位医生发表论文时的署名，在拼写上不尽相同，但是通过其传记资料，可以证明是同一个人。

2. Nathanael Fairfax, “Divers Instances of Peculiarities of Nature, Both in Men and Brutes;Communicated by the Same,” Philosophical Transactions 1666-67, 2(1666): 549-51.

3. 在瘟疫肆虐的那些年，牛顿有时也会在剑桥大学从事各种科学研究。关于牛顿的生平和著作，可以参考的书很多，例如：George Smith, “Isaac Newton,”The Stanford Encyclopedia of Philosophy, Fall 2008 edition, ed. Edward N. Zalta; V. Frederick Rickey, “Isaac Newton: Man, Myth, and Mathematics,” The College Mathematics Journal 18, no.5(1987): 362-89。

4. Tania Lombrozo, “Must Science Murder Its Darlings?” NPR 13.7: Cosmos and Culture, January 27, 2014.

5. Freeman J. Dyson, Infinite in All Directions, repr. ed.(New York: Harper Perennial, 2004; orig. pub.1988), 40.

6. Shane Parrish, “What Made Charles Darwin an Effective Thinker? Follow the Golden Rule,” Farnam Street, January 11, 2016. 除了科学领域，还有其他一些领域也拥有这种生物学倾向。例如，历史学家有时会有意远离抽象化和一般概念，以便让自己能更好地理解历史的复杂性。

7. 如果说生物学家和物理学家一样，也在拓展“实验还原主义”的极限，那么他们也会变得和工程师一样，沉迷于所研究系统的庞大规模和多样性，以及纯粹的复杂性。他们对“球形奶牛”这种概念完全没有兴趣。请参阅：John Doyle, “Computational Biology: Beyond the Spherical Cow,” Nature 411(2001): 151-52。

8. Steven A. Benner, “Aesthetics in Synthesis and Synthetic Biology,” Current Opinion in Chemical Biology 16, no.5-6(2012): 581-85.

9. 过时的遗留代码类似于软件中的“讨厌的遗迹”，对于当前版本的软件来说，这些东西其实是不必要的，但它们存在的时间却会比我们所希望的要久。

10. Whit Bronaugh, “The Trees That Miss the Mammoths,” American Forests, Winter 2010.

11. 过时的遗留代码不一定会被自然淘汰，但是可以通过对代码的定期整理和清扫来移除，这一点与生物学规律类似。

12. 乔治·戴森使用了另一个术语：数字生物学家，请参阅：“A Universe of Self-Replicating Code,” Edge.

13. “RNAi,” NOVA, July 26, 2005.

14. Howard Wainer and Shaun Lysen, “That's Funny,” American Scientist 97, no.4(2009): 272.

15. Dan Hurley, “Why Are So Few Blockbuster Drugs Invented Today?” The New York Times Magazine, November 13, 2014. 我们可以从药品的临床试验中学到很多东西，这一点是爱德华·荣格提醒我的。

16. Stewart Brand, The Clock of the Long Now: Time and Responsibility (New York: Basic Books, 1999), 85.

17. Peter G. Neumann, Computer-Related Risks (New York: ACM Press, 1995), 122.

18. Neal Stephenson, Cryptonomicon (New York: Avon Books, 1999; repr.2002), 802-3.

19. 需要注意的是，在《范布伦家的男孩们》（The Van Buren Boys）那一集中，酒吧里还有一个人也被称为“拉米雷斯”，不过我认为那个名字的发音明显不同于科尔基·拉米雷斯的“拉米雷斯”。也许，他当时就在那，但这只是我的猜测。《宋飞正传》的粉丝，请发邮件告诉我内情!

20. 也可以写为“Greebles”，请参阅：Kelly, What Technology Wants, 318。

21. Benoit B. Mandelbrot, The Fractal Geometry of Nature (New York: W. H. Freeman and Company, 1982), 1.

22. Borges, “Funes, His Memory,” in Collected Fictions, 131–37.

23. Philip Ball, Branches, vol.3 of Nature's Patterns: A Tapestry

in Three Parts (Oxford, UK: Oxford University Press, 2009), 181.

24. William Li et al., "Law Is Code: A Software Engineering Approach to Analyzing the United States Code," Journal of Business and Technology Law 10, no.2(2015): 297-372.

25. Jonathan Barnes, ed. and trans., Early Greek Philosophy (London: Penguin Classics, 1987); The Stanford Encyclopedia of Philosophy.

26. Barnes, Early Greek Philosophy, 72.

27. 这里关于"宇宙"和"本原"的观点都源于：Barnes, Early Greek Philosophy。

28. Ball, Curiosity, 98-99.

29. Lorraine Daston, "The Language of Strange Facts in Early Modern Science," Inscribing Science: Scientific Texts and the Materiality of Communication, ed. Timothy Lenoir(Stanford, CA: Stanford University Press, 1998), 20-38.

30. Daston, "Language of Strange Facts," 38.

31. Johan Bollen et al., "Clickstream Data Yields High-Resolution Maps of Science," PLoS ONE 4, no.3(2009): e4803.

32. Nicholas Donofrio, Jim Spohrer, and Hossein S. Zadeh, "Research-Driven Medical Education and Practice: A Case for T-shaped Professionals," MJA Viewpoint, 2009.

33. 对于专业化的深化发展趋势，霍默－狄克逊曾表示对"专家的狭隘化"十分担忧，请参阅：Homer-Dixon, The Ingenuity Gap, 176。

34. David J. Teece, "A Dynamic Capabilities Perspective on

Building Firm-Level Competitiveness," slide 43, Tusher Center on Intellectual Capital.

35. Nicola Griffith, Hild: A Novel (New York: Farrar, Straus and Giroux, 2013).

36. John Ptak, "A Cloud Map(1873)," JF Ptak Science Books, January 13, 2015.

## 06 生物学思维是理解复杂世界的一把金钥匙

1. Moses Maimonides, The Guide of the Perplexed, vol.1, trans. Shlomo Pines(Chicago: The University of Chicago Press, 1963), 65-66.

2. J. B. S. Haldane, Possible Worlds and Other Essays (London: Chatto &Windus, 1928), 286.

3. 关于"科学谦卑"的深入讨论，请参阅：Marcelo Gleiser, The Island of Knowledge: The Limits of Science and the Search for Meaning (New York: Basic Books, 2014)。

4. Ian Bogost, "The Cathedral of Computation," The Atlantic, January 15, 2015.

5. Bogost, "Cathedral of Computation."

6. Edsger Dijkstra, "The Humble Programmer." Communications of the ACM 15, no.10(1972): 859-66.

7. David Brooks, The Road to Character (New York: Random House, 2015), 263.

8. Carl Zimmer, "Is Most of Our DNA Garbage?" The New

York Times Magazine, March 5, 2015.

9. 正文中所引用的这些例句都来自盖尔所著作品的附录I：John Gall, The Systems Bible: The Beginner's Guide to Systems Large and Small, 3rd ed.(Walker, MN: The General Systemantics Press, 2003)。

10. 盖尔甚至建议他的读者“珍藏你的系统故障”，就像生物学家收集有趣的观察结果一样；他还以达尔文为例，强调我们必须承认“大型复杂系统是超出人类评估能力的”。请参阅：System Bible, xx, 68。

11. Robert Herritt, “When Technology Ceases to Amaze,” The New Atlantis 41, Winter 2014, 121-31.

12. Donald A. Norman, Living with Complexity (Cambridge, MA: The MIT Press, 2010), 117-18.

13. Daniel Engber, “Who Made That Progress Bar?” The New York Times Magazine, March 7, 2014.

14. Kate Greene, “How Should We Program Computers to Deceive?” Pacific Standard, September 3, 2014.

15. 菲利普·郭（Philip Guo）曾写道：“我想知道特波税务软件的代码中到底嵌套了多少if语句，”感谢丹·卡茨的提醒，特波税务软件可以被视为税法的一个界面。

16. Chaim Gingold, Miniature Gardens and Magic Crayons: Games, Spaces, and Worlds, master's thesis, Georgia Institute of Technology, 2003, 62.

17. Nicholas Carr, The Glass Cage: How Our Computers Are Changing Us (New York: W. W. Norton, 2014).

18. Winner, Autonomous Technology, 285.

19. Eytan Adar et al., “Benevolent Deception in Human Computer Interaction,” CHI’13: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,Paris, France, April 27—May 2, 2013(New York: ACM Digital Library, 2013): 1863-72.

20. 关于《模拟城市》的细节以及它对我们的启发，请参阅：Doug Bierend, “SimCity That I Used to Know: On the Game's 25th Birthday, a Devotee Talks with Creator Will Wright,” re: form, October 17, 2014。

21. Cowen, Average Is Over, 227–28．考恩的预言所针对的是经济学以及其他社会科学的未来，但在复杂性科学领域，同样需要能够解释未来社会的人。

22. 正如考恩所说，这些解释者在未来很可能会与机器通力合作。我们可能会喜欢上这种“人首马身”的怪物，当然，那不是半只马，而是半个机器。请参阅：Clive Thompson, Smarter Than You Think: How Technology Is Changing Our Minds for the Better (New York: Penguin, 2013)。

23. David Cope, “Experiments in Musical Intelligence.” Ryan Blitstein, “Triumph of the Cyborg Composer,” Pacific Standard, February 22, 2010.

24. 戴维·科普甚至在他的个人网站上写下了如下一段话，将这种“父与子”的自豪感扩展到了全人类的范畴：“算法所创作的音乐，与最具灵感的音乐家所创作的音乐一样，都是人类杰出的作品。”请参阅：Cope, “Experiments”。而且，他还说过这样的话：“计算机就是我自己的延伸。它们不是别的，就是一些非常好的铲子。我不会把挖好洞的功劳归于铲子，你会吗？”请参阅：Blitstein, “Triumph of the Cyborg Composer”。

25. 机器人专家汉斯·莫拉维克（Hans Moravec）把“更强大的后代”称为“心智儿童”(mind children)，在科幻作家姜峯楠（Ted Chiang）的一篇短篇小说中也有类似的情节设定。在这篇小说中，通过科技手段被强化的人，在科学探索方面的能力远远超过了普通人。最终，未被强化的，也就是普通人，则沦落到一无所知的地步。但是没关系，因为“我们用不着对‘超人’的科学成就感到恐惧。我们应该时刻牢记，构建出‘超人’的技术，最初是由人类开发的，所以它们并不比我们聪明”。请参阅：Luke Muehlhauser and Nick Bostrom, “Why We Need Friendly AI,” Think 36, no.13(Spring 2014), 41-47; Ted Chiang, Stories of Your Life and Others (New York: Tor Books, 2003), 203。

26. The World of Wonders: A Record of Things Wonderful in Nature, Science, and Art (London: Cassell, Petter, and Galpin, exact year of publication unknown).

27. 关于如何应对那些避重就轻的故事，请参阅：Philip Ball, “The Story Trap,” Aeon, November 12, 2015。

28. MLB. com, “Official Rules: 2.00 Definition of Terms.”

29. 这只是我个人对这两个术语的理解，毫无疑问，还有其他的区分方法。

30. Hubert Dreyfus and Sean Dorrance Kelly, All Things Shining: Reading the Western Classics to Find Meaning in a Secular Age (New York: Free Press, 2011), 88.

31. 关于如何克服对技术的无知，请参阅：Herritt, “When Technology Ceases to Amaze”。

32. “Customer FAQs Regarding the Sticking Accelerator Pedal and Floor Mat Pedal Entrapment Recalls,” Toyota Pressroom.

33. 例如，我们现在还无法完全理解自动驾驶汽车的所有技术细节，但这并不意味着，这种汽车比人类驾驶的汽车更危险。而且，对于人类驾驶的汽车，我们的理解也并不透彻，更不用说开车的人了。

34. 我要感谢提出这个建议的人，还要感谢堪萨斯大学哲学系的同事们，是他们为我提供了限制性定理的相关思路。

35. Carlin Romano, America the Philosophical (New York: Alfred A. Knopf, 2012), 501.

# 延伸阅读书目

本书正文未能呈现本人思想的全部来源。我在书中所阐述的思想，无论是来源还是基础，都很广泛。除了书中所提到的那些论著之外，我还想在这里强调一些引人深思的著作、论文、书籍和文章。在我撰写本书时，它们对我帮助不小。虽然有些著作和论文在正文中未被提及，但是如果你有兴趣深入研究本书所讨论的各个主题，那么这些参考文献应该都会很有用。

此外，限于篇幅，还有很多相关的重要主题没能在本书中被详细讨论到。它们包括：“技术变革的发生发展及其对教育的影响”，“大数据”，“人类与机器日益亲密的‘合作伙伴’关系”，“自动化趋势和未来就业之间的关系”，等等。另外，我们身边的各种不同类型的系统也很值得分析和讨论，例如制造业、食品业、政府部门、能源行业等。下面这些论著分析了上述主题与系统。

约翰·盖尔的《系统圣经》收录了一系列关于大型系统的奇闻轶事，皆与系统运行与系统故障有关。这是一本很有趣的书，我的许多想法都与盖尔的观点不谋而合。

凯文·凯利的《失控》阐释了很多生物学思维，并讨论了技术如何变得越来越生物化且无法被完全理解。该书观点与本书类似，不过是从涌现和生物复杂性，以及“如何使用生物学原理来构建技术”的角度来进行阐释的。此外，我还要推荐凯利的另一本著作《科技想要什么》（What Technology Wants）。

兰登·温纳的《自主的技术》，该书的倒数第2章特别值得一看，里面涉及的一些问题也正是本书所讨论的问题。

托马斯·霍默－狄克逊的《创造力差距》讨论了我们应该如何应对

日趋复杂和高深莫测的世界。该书还描述了各种各样的系统，其中有一些是本书未涉及的，它们都相当令人着迷。

查尔斯·佩罗（Charles Perrow）的《正常事故》（Normal Accidents）是一本关于系统失败和“如何与高风险技术共存”的经典论著。

维克拉姆·钱德拉的《极客升华》（Geek Sublime）对编程和计算机的本质进行了绝妙的反思，涉及了很多迷人的主题。

弗里曼·戴森的《全方位的无限》（Infinite in All Directions）很好地阐释了科学思想和发现的本质。

小弗雷德里克·布鲁克斯的《人月神话》是探讨软件开发与设计的经典作品。它重点讨论了工程师团队的管理问题，同时在软件和编程的性质方面提出了很多有趣的想法。

菲利普·鲍尔的《好奇心》（Curiosity: How Science Became Interested in Everything）非常出色地解释了科学家，特别是早期的科学家是如何探索世界，如何在极端多样化、极端繁杂的世界中寻找秩序感的。

唐纳德·诺曼的《与复杂性共存》（Living with Complexity）从设计的角度出发，考察了复杂性的起源，以及复杂化趋势的必然性。

布雷登·R.艾伦比（Braden R. Allenby）和丹尼尔·萨雷维茨（Daniel Sarewitz）共同撰写的《技术－人类条件》（Techno-Human Condition）探讨了我们应该如何应对行将到来的技术变革。它对“邪恶的复杂性”的分析特别有意思。

尼克·波斯特洛姆（Nick Bostrom）的《超级智能》（Superintelligence）探讨了与超智能机器发展有关的诸多问题。

凯特·阿舍尔的《建筑》（The Works）、《高度》（The

Heights）与《未来的方向》（The Way to Go）主要考察了城市、摩天大楼和交通网络，并分析了它们在现实世界中的运转方式，都是极具感染力的令人着迷的书。

埃里克·布林霍夫森（Erik Brynjolfsson）和安德鲁·麦卡菲（Andrew McAfee）合著的《第二个机器时代》（The Second Machine Age）考察了我们正在经历的、被寄予厚望的、高速发展的技术变革时代，并探讨了变革对经济的影响，以及我们应该如何应对等主题。

尼古拉斯·卡尔（Nicholas Carr）的《玻璃笼子》（Glass Cage）一书中阐述了自动化的危险性，以及我们身边的复杂技术。

马修·克劳福德（Matthew B. Crawford）的《摩托车修理店的未来工作哲学》（Shop Class as Soulcraft）[16]中探讨了重新“亲近”技术的重要性。亲力亲为，是美德的一部分。

由斯坦尼斯瓦夫·莱姆（Stanisław Lem）撰写，乔安娜·齐琳斯卡（Joanna Zylinska）翻译的《科技全书》（Summa Technologiae）讨论了自20世纪60年代以来，科技领域内出现的各种问题。这本出自未来科幻小说家之手的著作，特别强调了人类理解能力的极限。

迈克尔·莫布森（Michael Mauboussin）的《三思而后行》（Think Twice）讨论了如何才能对身边的复杂系统做出正确的思考。莫布森指出，正确的思考往往是违反直觉的。

罗伯特·赫里克（Robert Herritt）在《新亚特兰蒂斯》（New Atlantis）2014年冬季号，即总第41期，121—131页发表的《当技术不再令人惊奇》（When Technology Ceases to Amaze），是一篇讨论技术奇迹、复杂性和惊异感的优秀论文。

尼尔·斯蒂芬森的论文《最初……一切只是命令行》（In the

Beginning...Was the Command Line），后来扩展为一本同名小册子，详细地探讨了计算的协调问题。尽管书中的观点在现在看来已经有点过时了，但在关于“人类如何与技术相联系，又如何与技术分离”这个方面，还是颇具见地的。

伦纳德·E.里德（Leonard E. Read）的《铅笔的故事》（I, Pencil）是一篇隽永的短文，它探讨了制造一支铅笔所涉及的高度互联的社会经济体系。对于这个系统，任何人都无法完全理解。

我还建议读者阅读一下艾兹格·迪科斯彻的其他论著。他的许多作品都是计算机科学领域的经典，同时也涉及了有关技术本质的哲学思考。

如需更多阅读材料，请访问我的个人网站，查找相关主题的论文。

# 致谢

感谢我的编辑妮基·帕帕佐普洛斯（Niki Papadopoulos），她对本书的写作给予了宝贵的支持。感谢马克斯·布罗克曼（Max Brockman），他帮我提炼了本书的核心思想。

本书的部分内容曾在其他地方发表过，包括《万古》（Aeon）、《鹦鹉螺》（Nautilus）、《页岩》、《连线》等杂志，以及Edge网站出品的《如何思考会思考的机器》（What Do You Think About Machines That Think?）一书。我为能有机会公开阐述个人思想而表示感谢。我还要感谢《连线》杂志的科学栏目邀请我开通了博客，我在本书中提到的不少想法都曾以更为不连贯的形式在博客上发表。此外，感谢我电子邮件简报的订阅者们，他们也提前读到了本书的若干内容。

感谢我所有的读者和对谈者，感谢大家慷慨付出的大量时间和丰富的专业知识，在今天这个时代，这两者都非常珍贵。他们是：乔希·阿贝斯曼（Josh Arbesman）、泽夫·伯杰（Zev Berger）、安德鲁·布卢姆、亚伦·克劳塞特（Aaron Clauset）、洛丽·埃默生（Lori Emerson）、乔舒亚·费尔菲尔德（Joshua Fairfield）、亨利·法雷尔（Henry Farrell）、劳伦斯·冈萨雷斯（Laurence Gonzales）、爱德华·荣格、亚伦·卡恩（Aaron Kahn）、丹·卡茨（Dan Katz）、迈克尔·科兴德费尔（Mykel Kochenderfer）、史蒂文·米勒（Steven Miller）、梅根·欧文（Megan Owen）、埃尔纳坦·赖斯纳（Elnatan Reisner）、内厄姆·沙尔曼（Nahum Shalman）、雅各布·舍曼（Jacob Sherman）、特德·斯坦伯格（Ted Steinberg）、布赖恩·斯蒂芬斯（Brian Stephens）、哈里·萨尔顿（Harry Surden）和杰文·韦斯特（Jevin West）。上面这些人，以及其他一些人，帮助我对本书进行了修订，并避免了一些相当

荒谬的错误；也就是说，书中若还存在错误，责任都在我自己身上。

我还要特别感谢堪萨斯大学哲学研究小组的所有成员。该小组致力于考察人类的理解能力与软件之间的关系。作为堪萨斯大学哲学系的一名访问学者，我也是这个小组的成员之一。小组成员约翰·西蒙斯、杰克·霍纳和拉蒙·阿尔瓦拉多（Ramon Alvarado）的意见和建议，对本书而言具有无法估量的价值。戴维·斯蒂恩（David Steen）为我提供了有关野生动物生态学和野外生物学方面的指引；乔希·森夏恩（Josh Sunshine）在计算机科学和软件工程方面给了我大量的支持和建议；迈克尔·维特维奇（Michael Vitevitch）帮助我厘清了语言的本质和人类处理语言的方式等方面的问题；迈克尔·巴尔和菲利普·库普曼就丰田汽车意外加速事故和技术的复杂性问题，为我提供了有益的建议。

感谢尤因·马里恩·考夫曼基金会（Ewing Marion Kauffman Foundation），它为我创造了一个很好的写作环境。感谢基金会的同事们，感谢你们无与伦比的热情和快速的反馈，特别感谢你们的大度，允许我按自己的进度完成本书。

此外，拉克斯资本（Lux Capital）的项目团队也对本书的写作给予了极大的支持，并为我提供了大量的建议和意见。非常感谢大家。

最后，感谢我的家人。我的父母在第一时间为我提供了支持和建议，并针对初稿给出了详细意见。祖父是我最出色的听众。我非常感谢家里的每一个人。特别感谢我的妻子德布拉（Debra），是你给了我空间和时间，让我可以保质保量地完成这本书。在整个写作过程中，从创作初稿开始，你的不急不躁一直令我惊叹不已。

感谢阿比高尔。你是那么喜欢读书，尽管现在的你更喜欢看那些文字简单、色彩炫丽的书。你每天都会为我带来无限欢乐。从第一行字开始，你一直支持我写这本书，尽管你的支持主要表现为热情的拥抱、无拘无束的欢笑和快乐的闲聊，但是这些已经比我所期望的要多得多了。

虽然我也希望能有众多的读者喜欢这本书，但它终究是为你和你的弟弟内森写的。你们俩注定要在一个日益复杂的世界中长大成人。未来世界会是什么样，现在的我只能靠想象。不过，我敢肯定的是，它与我们今天所处的世界大为不同，它将变得更加难以理解。我希望通过这本书，以一种举重若轻的方式让你明白，为什么你们不用为那样的未来感到害怕。

# 译者后记

在这个越来越复杂的世界上，我们已经无法理解自己创造出来的系统了。技术系统、经济系统、政治系统、法律系统，莫不如是。这是一个悖论。这些过于复杂的系统，在使我们的生活更加便利的同时，本身却变得无法把控了；更加严重的问题是，我们无法预知，这些系统会在什么时候突然崩溃，造成不可承受的损失。

从技术的角度来看，这种复杂性产生于为了适应边界情况而不断加入补丁的过程。任何一个人造系统，哪怕最初是根据清晰的数学模型构建，用于非常明确的用途，只要一直被使用，就肯定会成为一个补丁加补丁的拼凑起来的系统。这是一个进化过程，让人造系统越来越远离确定性。

那么应该怎么办呢？塞缪尔·阿贝斯曼告诉我们，虽然我们应该努力去理解人类无法理解的复杂系统，但首先必须接受混乱。只有这样，我们才能静下心去观察各种意想不到的事件，获得关于我们所用的算法是怎样真正起作用的线索。他强调，要认识世界的复杂性，必须要结合物理学和生物学：用生物学思维处理个例，用物理学思维提取规律。总之，要承认复杂性的存在，同时不要放弃理解系统的原理，而且要从简单的组件开始。

塞缪尔·阿贝斯曼在这本书中给我们提供了应对超复杂这个难题的一些方法，但是并没有提供终极答案。事实上，这个问题也不应该有终极答案。这是因为，复杂是人类行动的非意图后果的集中反映。一方面，在进入现代社会后，我们的任何一个行动，都可能产生比以往任何时候都意想不到的后果；另一方面，我们对问题的暂时解决，反而可能掩盖了问题的根本原因。但是，这种“困境”可以说是不可避免的。它虽然有时会令我们承受痛苦，但却可能是人类进步最主要的途径。它也

告诉我们，一劳永逸解决问题的可能性从来就是微乎其微的。这一点对公共政策领域的决策者尤其重要。

说到底，思考和应对复杂问题，需要的是思维，以及思维的多样性。一言以蔽之，真正重要的是，我们要努力让自己成为创新者、思想家和科学家。

这本书的翻译得以完成，要感谢很多人的支持和帮助。

我首先要感谢的是我的太太傅瑞蓉。我的每一本译著，她都有一大半的功劳。同时也要感谢儿子贾岚晴带给我的动力和快乐。

感谢有情怀、重情义、善创新、爱读书的农夫山泉董事长钟睒睒，他既是好老板，也是好老师，让我在工作之余能够腾出时间来完成此书的翻译。

还要借此机会感谢汪丁丁教授、叶航教授和罗卫东教授的教诲。我对复杂性科学的兴趣源于他们。同时还要感谢何永勤、虞伟华、余仲望、鲍玮玮、傅晓燕、傅锐飞、陈叶烽、童乙伦、罗俊、邓昊力、陈姝、黄达强、李燕、李欢、丁玫、何志星等好友的支持和帮助。

感谢湛庐文化。感谢简学。

书中错漏之处在所难免，敬请专家和读者批评指正。

贾拥民

于杭州崇谷阁

# 未来，属于终身学习者

我这辈子遇到的聪明人（来自各行各业的聪明人）没有不每天阅读的——没有，一个都没有。巴菲特读书之多，我读书之多，可能会让你感到吃惊。孩子们都笑话我。他们觉得我是一本长了两条腿的书。

——查理·芒格

互联网改变了信息连接的方式；指数型技术在迅速颠覆着现有的商业世界；人工智能已经开始抢占人类的工作岗位……

未来，到底需要什么样的人才？

改变命运唯一的策略是你要变成终身学习者。未来世界将不再需要单一的技能型人才，而是需要具备完善的知识结构、极强逻辑思考力和高感知力的复合型人才。优秀的人往往通过阅读建立足够强大的抽象思维能力，获得异于众人的思考和整合能力。未来，将属于终身学习者！而阅读必定和终身学习形影不离。

很多人读书，追求的是干货，寻求的是立刻行之有效的解决方案。其实这是一种留在舒适区的阅读方法。在这个充满不确定性的年代，答案不会简单地出现在书里，因为生活根本就没有标准确切的答案，你也不能期望过去的经验能解决未来的问题。

## 湛庐阅读APP：与最聪明的人共同进化

有人常常把成本支出的焦点放在书价上，把读完一本书当作阅读的终结。其实不然。

时间是读者付出的最大阅读成本
怎么读是读者面临的最大阅读障碍
“读书破万卷”不仅仅在“万”，更重要的是在“破”！

现在，我们构建了全新的 “湛庐阅读”APP。它将成为你“破万卷”的新居所。在这里：

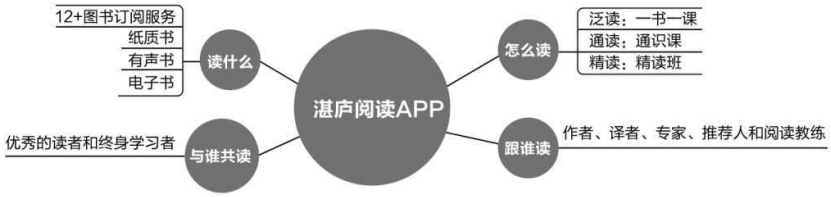
- 不用考虑读什么，你可以便捷找到纸书、有声书和各种声音产品；
- 你可以学会怎么读，你将发现集泛读、通读、精读于一体的阅读解决方案；
- 你会与作者、译者、专家、推荐人和阅读教练相遇，他们是优质思想的发源地；
- 你会与优秀的读者和终身学习者为伍，他们对阅读和学习有着持久的热情和源源不绝的内驱力。

从单一到复合，从知道到精通，从理解到创造，湛庐希望建立一个“与最聪明的人共同进化”的社区，成为人类先进思想交汇的聚集地，与你共同迎接未来。

与此同时，我们希望能够重新定义你的学习场景，让你随时随地收获有内容、有价值的思想，通过阅读实现终身学习。这是我们的使命和价值。

# 湛庐阅读APP玩转指南

## 湛庐阅读APP结构图：




## 三步玩转湛庐阅读APP：


APP获取方式：

安卓用户前往各大应用市场、苹果用户前往APP Store
直接下载“湛庐阅读”APP，与最聪明的人共同进化！

# 使用APP扫一扫功能，
# 遇见书里书外更大的世界！


快速了解本书内容，
湛庐千册图书一键购买！

大咖优质课、
献声朗读全本一键了解，
为你读书、讲书、拆书！

你想知道的彩蛋
和本书更多知识、资讯，
尽在延伸阅读！

湛庐CHEERS

# 延伸阅读

## 《技术的本质》经典版

◎ 复杂性科学的奠基者、著名的技术思想家、“熊彼特“奖的得主布莱恩·阿瑟的经典作品。

◎ 埃里克·施密特、凯文·凯利、汪丁丁、段永朝、陈劲、包国光等联袂推荐。

◎ 技术理论体系的先河之作，独具创新的关于技术产生和进化的系统性理论！一次打开“技术黑箱”的尝试性创新探索！


使用“湛庐阅读”APP，
“扫一扫”获取本书更多精彩内容。

ISBN 978-7-213-05998-8

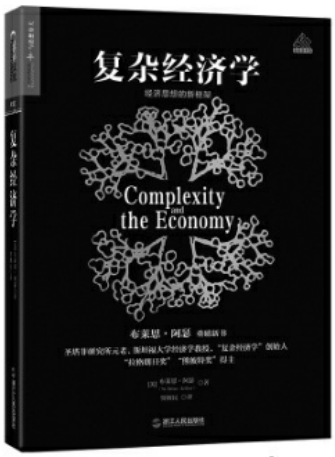
## 《复杂经济学》

◎ 作为“复杂经济学”的创始人，布莱恩·阿瑟在本书中汇集了多年对复杂经济学的研究。其核心思想可以归结为：经济不一定处于均衡状态，演绎推理将被归纳推理所取代。

◎ 这是一本见证“复杂经济学”成长的著作，你将看到“复杂性思

维”在经济学领域是如何发展起来并形成一门崭新的学科。

◎ 详细讨论了“爱尔法鲁酒吧问题”和“圣塔菲人工股票市场”两个重要项目。




使用“湛庐阅读”APP，
“扫一扫”获取本书更多精彩内容。
ISBN 978-7-213-08645-8
9 787213 086458

## 《多样性红利》

◎ 一个人是否聪明不是由智商决定的，而取决于认知工具的多样性！

◎ 《多样性红利》创造性提出多样性视角、启发式、解释和预测模型四个认知工具箱框架。

◎ 广受欢迎的“模型思维课”主讲人斯科特·佩奇经典名作！


使用“湛庐阅读”APP，
“扫一扫”获取本书更多精彩内容。
ISBN 978-7-5536-7385-1
9 787553 673851

# 《同步：秩序如何从混沌中涌现》

◎ “复杂性理论”开创性研究者斯蒂芬·斯托加茨重磅新作！

◎ 互联网时代最重要的观念之一“小世界网络“的提出者！

◎ 电子科技大学教授周涛，诺贝尔奖获得者菲利普·安德森，牛津大学教授罗伯特·梅，美国海军研究实验室卢·佩科拉等联袂推荐！


使用“湛庐阅读”APP，
“扫一扫”获取本书更多精彩内容。
ISBN 978-7-220-10695-8
9 787220 106958 >

---

(1) 即隐藏的代码，通常用于实现某些系统功能或提示某些界面信息等。——编者注

(2) 意为被设计得过度复杂的机械组合，以迂回曲折的方法完成一些其实非常简单的工作。——编者注

(3) 指将数据直接嵌入到程序或其他可执行对象的源代码中的软件开发实践。——编者注

(4) 幂律来自20世纪20年代对英语单词使用频率所做的分析，语言学家发现单词使用的频率和它的使用优先度是一个常数次幂的反比关系。它拥有两个通俗化的解释，一个是长尾理论，另一个是马太效应。——编者注

(5) 又称拉普拉斯－高斯曲线，或正态分布曲线，两端低中间高，常被数学家用来描述科学观察中量度与误差的分布。——编者注

(6) 史蒂芬·霍金在《时间简史》里讲过这样一个故事：一位科学家在演讲结束时被一位老妇人讥讽：“你说的都是废话。这个世界不过是一个乌龟背上的平板。”科学家问：“那么，这只乌龟是站在什么上的呢？”老妇人说：“当然

是乌龟群啊！”——编者注

(7) 这些应用程序的功能大体相同，但是运行方式不同，例如微软系统中的PowerPoint与苹果系统中的Keynote。

(8) 史蒂芬·平克的著作《当下的启蒙》《思想本质》《白板》《语言本能》《心智探奇》已由湛庐文化策划，浙江人民出版社出版。——编者注

(9) 爱德华·威尔逊的著作《创造的本源》《半个地球》等，已由湛庐文化策划，浙江人民出版社出版。——编者注

(10) 研究“生物各层次结构与功能的关系”“生命活动的物理”“物理化学过程”和“物质在生命活动过程中表现的物理特性”的生物学分支学科。——编者注

(11) 在管理工作中运用线性规划、非线性规划、动态规划和整数规划，以及系统科学方法所确定的表示最优方案的模型。——编者注

(12) 乔治·戴森的著作《图灵的大教堂》已由湛庐文化策划，浙江人民出版社出版。——编者注

(13) 是一种概率分布模型，包含长尾分布和次指数分布两种子类型。——编者注

(14) 即失去原本矮壮的造型，茎叶疯狂伸长的现象。主因是缺少日照，或光线过暗，同时浇水又相对较多。——编者注

(15) 美国环保主义动画影集《地球超人》中的人物。——编者注

(16) 本书已由湛庐文化策划，浙江人民出版社出版。——编者注