文章编号 2095-0020(2020)01-0044-06

对伪随机数生成算法的随机性评价方法的研究

丁豪杰[1,2]， 唐 迪[3]， 姚 琳[4]， 顾幸生[2]

（1. 上海立信会计金融学院 校长办公室，上海 201209；2. 华东理工大学 化工过程先进控制和优化技术教育部重点实验室，上海 200237；3. 云南师范大学文理学院，昆明 650222；4. 上海第二工业大学 工学部，上海 201209）

摘 要 在基于种群的元启发式算法的实现中，随机数的生成是一个无法忽视和回避的问题，而目前的算法程序主要基于冯·诺依曼体系架构的计算机实现。在据冯·诺依曼机架构体系下，真正的随机数只能依靠伪随机数生成器算法程序产生的伪随机数序列来逼近真实的随机数。由于伪随机数的随机性决定了伪随机数生成器的性能优劣，因此，对伪随机数生成算法的随机性评价十分重要。提出了一种评价伪随机数的随机性的新方法，在评价伪随机数生成器算法性能的同时还可帮助指导其进一步改良。

关键词 生产调度；制造；群智能算法；冯·诺依曼机架构；伪随机数

中图分类号 TP 301 **文献标志码** A

Evaluation research on the randomness of pseudo-random number generator algorithms

DING Haojie[1,2], *TANG Di*[3], *YAO Lin*[4], *GU Xingsheng*[2]

(1. President's Office, Shanghai Lixin University of Accounting and Finance, Shanghai 201209, China; 2. Key Laboratory of Advanced Control and Optimization for Chemical Processes, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China; 3. The College of Arts and Sciences, Yunnan Normal University, Kunming 650222, China; 4. College of Engineering, Shanghai Polytechnic University, Shanghai 201209, China)

Abstract Random number generators cannot be neglected to metaheuristics based on populations. The present algorithm programs are implemented based on the architecture of the von Neumann machine. Under the architecture, the real random number generators cannot be programmed on the von Neumann computers, but can be approached by the pseudo-random number permutation, which is generated by a pseudo-random number generator. Since the randomness decides the pseudo-random number generator performance, evaluating the randomness of the

收稿日期：2019-12-24

作者简介：丁豪杰(1978—)，男，工程师，主要研究方向为控制理论与控制工程及生产调度算法和组织管理优化等，E-mail：hero_ding1978@163.com

pseudo-random number generator algorithms is important. The present work proposes a novel evaluating method to the randomness of pseudo-random number generator algorithms and can conduct the pseudo-random number generator algorithm improvements.

Key words　production scheduling; manufacturing; swarm intelligent algorithm; architecture of Von Neumann machine; pseudo-random number

目前较为流行的启发式算法中，有一类基于种群的元启发式算法，对于解决诸如生产调度等被证明属于 NP-complete 或 NP-hard 问题的实际问题已经取得了不俗的成果[1]。这类基于种群的元启发式算法被称为群智能算法[2]。

调度是使用一定的基本规则在众多制造业和服务业中进行决策的过程，它研究在给定的时间周期内将稀缺（或有限）的资源分配给待执行的任务，目标是优化一个或多个目标[3]。以汽车等机械加工制造企业的生产为例，企业的生产调度任务就是优化资源分配，合理分配生产计划任务[4]。当生产加工和整车装配过程中出现未预期事件，如生产设备故障、某些零部件的加工时间异常、部件原材料供应发生问题导致无法及时生产等问题，都要及时予以处理，以保证企业的生产能正常进行[5]。类似的调度问题处理都可以转化成为对随机优化问题的处理。

运用群智能算法借助计算机处理此类随机优化问题时，往往采用全局搜索与局部搜索相结合的方法进行，在开始阶段以全局搜索为主，而在后续阶段则以局部的邻域搜索为主。群智能算法在搜索过程中，需要大量的随机数对算法进行支持，故随机数的选取成为一个不可忽视和回避的问题。在全局搜索阶段，为不遗漏可能的解，应尽可能使与随机数相对应的解充满整个解空间，宜选用符合均匀分布的随机数；而在后续有趋势解呈现时，则可以选择符合某种分布的随机数来加速求解，后者主要靠前者与经验及规则函数配合完成。由于目前启发式算法主要借助于计算机来完成，因此，使用计算机来产生随机数从而完成相应的求解是必要的步骤。但由于目前的主流计算机设计架构尚未突破冯·诺依曼体系架构[6]，真正的随机数是无法在冯·诺依曼机上产生和实现的（参阅下文之基本概念），只能依靠采用以伪随机数算法指导的程序生成伪随机数序列来进行近似逼近。因此，一个尽量随机的伪随机数算法是必要的。这种指导程序生成伪随机数序列算法的随机性决定了该算法的性能。正确的评价一个伪随机数生成算法的性能，对指导实际求解问题的开展具有决定意义。

本文以运用群智能算法求解生产调度问题为例，对正确评价伪随机数生成算法进行深入探讨和研究。

1　基本概念

1.1　随机数

随机数是指一组不能被预测的数字或符号序列，这些数字和序列是完全偶然产生的[7]，即随机数是客观世界不确定性的理论抽象[8]。随着概率论及数理统计等学科的发展，研究人员对随机数的产生机理等逐渐明晰。同时，为了研究的深入进行，从软硬件等方面尽可能设计了多种计算机实现方法。但是，依据冯·诺依曼架构设计的计算机在不借助于外部硬件，仅从软件自身角度，真实的随机数是无法实现的。因为冯·诺依曼机自身的架构决定了这种机型只能做“按存储指令的顺序来执行具体指令”的确定性行为。

1.2　冯·诺依曼机和冯·诺依曼架构

冯·诺依曼机是指使用冯诺依曼体系架构的电子数字计算机。早在 1945 年 2 月，世界上第一台电脑诞生前，冯·诺依曼就提出了在数字计算机内部的存储器中存放程序的概念，正是按照冯·诺依曼的设计理念，在美国的宾夕法尼亚大学设计实现了第一台计算机 ENIAC[9]，用以计算弹道轨道。这是当今主流计算机的模板，被称为冯·诺依曼结构[10]，目前，所有在用的成熟计算机设备均采用冯·诺依曼结构设计。冯·诺依曼机主要由逻辑运算单元、控制器、存储单元、输入输出设备和其间相关总线组成，主要特点是：① 程序经编译后形成

固定顺序的指令，以二进制代码的形式存放在存储器中；② 所有的指令都是由操作码和地址码组成；③ 指令按照存储的顺序调入内部的堆栈，并进行调用；④ 以逻辑运算单元和控制器作为计算机处理的中心，并依次执行指令。

由上述冯·诺依曼机设计架构的特点可以推断出：其内部的每一个核心在调用和执行指令时是一种串行机理，执行结果只依赖于存储的程序和编译生成的指令代码调用次序。因此，依据冯·诺依曼机架构设计制造的计算机只能精确而有序地实现程序指令，不具备随机不确定性，在不引入外部随机量的情况下，单纯依靠精确而有序地执行指令的冯·诺依曼机自身来产生真随机数，本身就是一个悖论。因此，只能使用伪随机数来逼近真实的随机数。

1.3 伪随机数和伪随机数生成器

伪随机数则是指一组特性接近随机数特性的数的序列，多由伪随机数生成器产生[11]。伪随机数生成器也被称为确定的随机二进制数位生成器，是生成一组伪随机数的算法。由上述定义及相关研究[12]可知，伪随机数实际是确定的数的序列，只是其序列周期的长度远长于所研究的问题集分布序列的长度。而在基于冯·诺依曼架构的计算机上，不依靠外部设备引入随机性，只能通过计算机程序产生伪随机数序列，该程序即相当于伪随机数生成器。

2 伪随机数评价方法构建

2.1 伪随机数算法评价的必要性

当今，各领域所使用的主流计算机均未突破冯·诺依曼机的设计理念。各领域若仅借助于计算机进行研究，其所需的随机数只能通过伪随机数生成器程序所生成的伪随机数序列逼近。如何选择适用于待解问题的已有伪随机数生成器程序，或如何改进已有的伪随机数生成算法，编制适合于待解问题的伪随机数生成程序具有重要意义。正确的选择或改进已有的伪随机数生成算法就必须要对待研究的伪随机数算法进行正确的评价。根据没有免费午餐定理[13]，任何相对较优的算法，不可能对该领域的所有问题均产生同样显著的效果；其只会在某些问题上产生较优的效果，而在另一些问题上则不会产生显著效果。因此，问题不同，指导程序设计的伪随机数算法就可能不同，正确评价才能针对特定问题产生有效的随机数。

2.2 新评价方法的设计诱因

对随机数性能的评价主要基于对其测试的基础之上。现有的主要测试方法是使用美国国家标准与技术研究院(National Iustitute of Standards and Technology，NIST)随机性测试方法中的一种或几种组合，NIST 随机性测试方法主要包括：频数测试、块内频数测试、游程测试、块内最长连续“1”测试、矩阵秩测试、离散傅里叶变换测试、非重叠模板匹配测试、重叠模板匹配测试、通用统计测试、压缩测试、线性复杂度测试、连续性测试、近似熵测试、部分和测试、随机游走测试及随机游走变量测试等[14]。

由于传统伪随机数生成器的测试方法过于关注其数学性能，即关注于评价其是否符合某些随机分布的特性，而忽略了其工程应用方面的实际需求，使得设计的算法过于复杂。根据自身领域经验，针对符合均匀随机分布的伪随机数生成器的测试，本文提出了新的测试方法，并用此测试方法对伪随机数的性能进行了评价，从而达到了对所测试的伪随机数算法进行正确评价的目的。

2.3 新评价方法的设计实现

由于研究的数字应服从均匀分布，其数值中的各个权重位上的数字均应呈现出均匀随机特性，即在 0～9 的 10 个数字中均匀产生。据此，可以将所需精度的数字一分为二，作为类似位图填充的横、纵坐标点来进行考量，即在前半部分和后半部分数字形成的平面坐标点所对应的位置进行填充。具体实施时，可以借助二维 0-1 矩阵来较为简捷地实现。此时随机数的前半部分和后半部分数字对应着矩阵具体的行数和列数。

以 0 到 1 间服从均匀分布的 2 位有效数字的伪随机数生成器为例，此时可以构建 10×10 的初始零矩阵 M，如果生成器产生了伪随机数 0.34，则对应将 $M(3,4)$ 置 1，此时可以利用相关软件对此 0-1 矩阵绘图，将矩阵值为 1 的方格进行填充，如

图 1 所示(为突出显示效果，应尽量用深色表示矩阵中较少的颜色)。最终在执行若干次类似操作后，M 矩阵的相应元素将会被陆续置 1，此时绘出的填充示例图可以直观表示出所产生随机数的分布状态。考虑到随机数生成器的数学特性，测试执行的次数应与矩阵元素的个数相等；否则，从概率角度审视，过多的执行次数会使整个矩阵绝大部分元素置 1，一方面无法考察其随机特性，另一方面使相应绘出的矩阵填充图失去其应有的直观性。另外，为了直观起见，选择测试有效位数时，应尽可能选择偶数位数的伪随机数，以便于对等拆分，并在拆分之后以前后两个数字作为坐标生成相应的方阵和绘制相应的测试结果示例图。

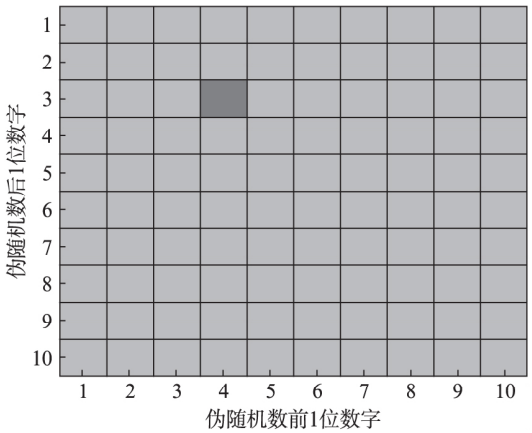
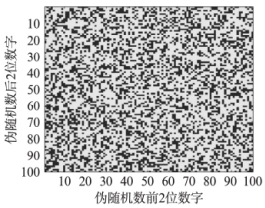
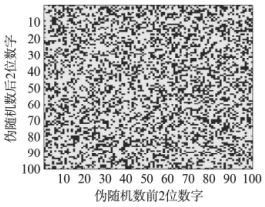
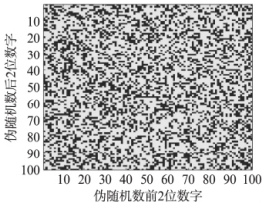
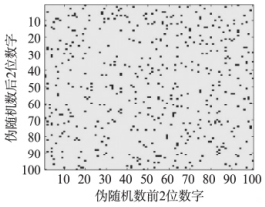


图 1　示例说明

3　新测试评价过程实例

本文选择了 3 种伪随机数算法指导下编制的程序，即目前程序设计中常用的 C 语言自带的 rand()随机函数，Kiss 随机函数和 Mersenne 随机函数[15]。测试在 SUSE Linux Enterprise 12 软件虚拟机平台上进行，使用 C 语言编程调用相关函数进行测试，并借助 Matlab 2015b 软件对生成的矩阵进行填充图绘制。考虑到位图较小，为了增强图像色差效果，便于对比观察，绘图时省略了网格线，并且用浅色表示矩阵中较多的元素“1”，用深色表示矩阵中相对较少的元素“0”。执行次数定为 10 000 次(100×100)，对初始零矩阵的覆盖率(最终矩阵中 1 的元素个数与矩阵总元素的百分比)，及首次达到 60%覆盖率所执行的次数进行了考察，结果如表 1 所示。首次达到 60%覆盖率直观显示了算法的分散度或随机性。

表 1　测试结果对比

矩阵序号	伪随机函数	覆盖率/%	覆盖率首次达到 60%时的执行次数	矩阵填充图
M_1	rand()(C 语言函数库自带)	63.96	8 998	
M_2	rand_kiss()函数(依据 KISS 算法编制)	63.55	9 065	
M_3	rand_mersenne()(Mersenne 算法)	62.99	9 228	
M	—	95.09	—	

* 矩阵 M 为 M_1、M_2 与 M_3 运算合成，而非程序生成，故“覆盖率首次达 60%的执行次数”处填“—”。

从表 1 中可以看出，3 种方法各方面数据相差并不显著，但是所示的矩阵填充图却直观给出了 3 种函数具体的数值分布情况。仔细观察 3 张填充图，可以发现 3 个伪随机函数生成器所产生的伪随机数分布是有错落的。据此，对 3 张图进行类似位图的异或逻辑运算，即比对 3 个矩阵，凡是对应位置元素相异的则置 1，最终生成矩阵 M 如图 2 所示($M=M_1\oplus M_2\oplus M_3$)，其覆盖率和对应的矩阵填充图列于表格的最后一行。也有了产生新伪随机数方法的一种启示，即依靠机理不同的伪随机数算法简单加权组合生成新的伪随机数算法。

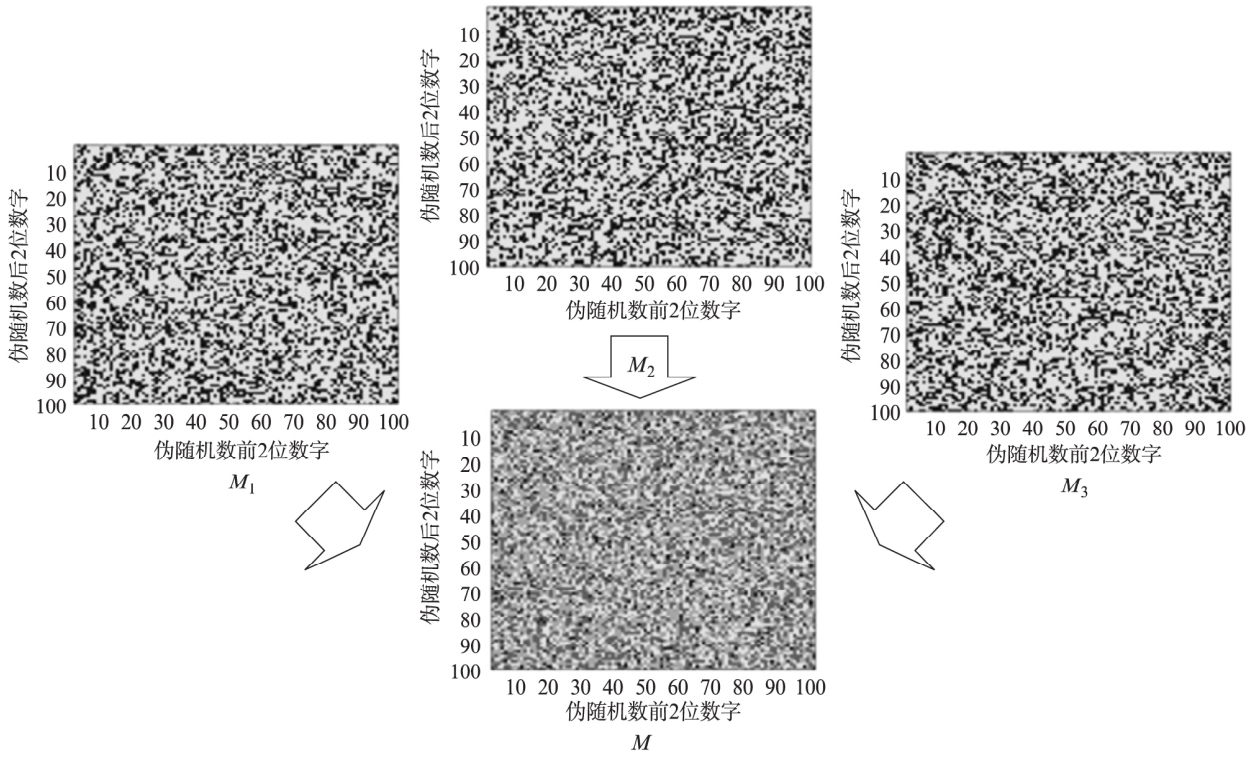


图 2 M_1、M_2、M_3 矩阵重叠为 M 矩阵

4 实例分析和讨论

4.1 实例分析

由表 1 可以看出，最后一行组合后的矩阵填充图的效果最好，其覆盖率达到了 95.09%，而单独的每一个伪随机函数生成器所产生的随机数覆盖率只有 63%左右。这主要是合成图相当于每次填充时进行了 3 次随机填充，即执行了原迭代次数的 3 倍(30 000 次)。为统一比较单独伪随机数函数与组合伪随机数函数的优劣，本文以上述 3 个伪随机函数为基础，计算所得数据平均值如表 2 所示。

表 2 中使用了简单的等权组合各种伪随机数算法。由表 2 不难看出，对伪随机函数算法进行不同组合，有可能会指导产生效果更好的伪随机数生成器，如同样进行 2 次填充，M_{12} 伪随机数生成器覆盖范围更大。这启示当采用不同的组合加权系数时，有机会生成效果更适合的伪随机数生成器。

4.2 测试评价过程理论分析

对比表 1 和表 2 可以发现：每种方法的覆盖率与函数无关，只与函数的个数有关，其原因可以用离散性随机变量的概率进一步证明。论证如下：

表 2 函数组合测试结果

矩阵序号	伪随机函数	平均覆盖率/%	覆盖率首次达到60%时的平均执行次数
$M_1\times2$	rand()×2	86.58	4 499
$M_2\times2$	rand_kiss()×2	86.81	4 532
$M_3\times2$	rand_mersenne()×2	86.38	4 614
M_{12}	rand()+rand_kiss()	86.87	4 526
M_{23}	rand_kiss()+rand_mersenne()	86.20	4 528
M_{31}	rand_mersenne()+rand()	86.50	4 617
$M_1\times3$	rand()×3	95.16	2 999
$M_2\times3$	rand_kiss()×3	95.22	3 021
$M_3\times3$	rand_mersenne()×3	94.92	3 076
M	rand()+rand_kiss()+rand_mersenne()	95.09	3 089

说明：表中“×2”表示每次填充时进行了 2 次随机填充，“×3”表示每次填充时进行了 3 次随机填充。

上述每次绘点的过程是完全独立的，而点的位置也是随机的，因此，该过程可以看成是符合伯努

利二项式分布的。其可以对应为初始零矩阵中的元素被选中置 1 的事件过程。

设随机变量 X 表示：初始零矩阵中的元素被选中置 1 的事件，则 $X\sim B(n,p)$。其中，

$$n=10\,000,\quad p=1/10\,000$$

由于此时 n 很大，且 p 很小，故可近似认定 X 服从泊松分布，即 $X\sim P(\lambda)$。其中，

$$\lambda=np=1$$

因此，初始零矩阵每次置 1 的事件概率为

$$P(X)=P\{X=1\}+P\{X=2\}+\cdots+P\{X=10\,000\}=1-P\{X=0\}=1-e^{-1}=63.21\%$$

由此证明可以推知：表 1 中的初始零矩阵覆盖率维系在 63%附近是合理的；选取“覆盖率首次达到 60%的执行次数”来体现函数的速度也是恰当的；同时还证明实验中所选的伪随机函数生成器都具有良好的数学特性，是值得信赖的。当进一步使用两种随机函数混合运算时，并不是简单对两个函数求均值，而是在一次置 1 的事件过程中依次使用随机函数对初始零矩阵置 1。

再设所选用的每一个伪随机函数具有上述数学特性且保持独立，函数个数为 N。则上述每次被选中置 1 的概率为：$p=N/10\,000$，相应的 $\lambda=np=N$。由此可得函数个数与覆盖率的函数关系，即覆盖率 $CN=1-e^{-N}$。表 3 所示列出了 N 取 1～5 时的理论覆盖率。

表 3　1～5 个函数时的覆盖率

N	理论覆盖率/%	rand()测试覆盖率/%	覆盖率首次达到60%时的执行次数
1	63.21	63.96	8 998
2	86.47	86.58	4 499
3	95.02	95.16	2 999
4	98.17	98.23	2 249
5	99.33	99.30	1 799

由表 3 中的理论覆盖率可以看出，使用独立的随机函数越多，覆盖效果越好。而从实际执行层面看，程序在一次置 1 事件的执行过程中，实际又独立执行了 N 次置 1 事件，增加了置 1 概率，从而增加了最终的初始零矩阵覆盖率。

4.3　实例的指导意义

根据上述分析可以看出，既然所选的伪随机函数生成器相互间独立，其置 1 过程也是独立的，多次独立使用同一个伪随机函数生成器来达到多个伪随机数生成器的效果也可作进一步的研究。笔者利用 3 个函数中最简单的伪随机函数 rand()，在程序执行一次置 1 的过程中，进行了 N 次独立随机使用，所得数据列于表 3 的最后两列。由数据看，同样达到了组合随机函数的效果，随着使用次数的增加，覆盖率逐渐接近于 1，且“覆盖率首次达到 60%时的执行次数”也显著降低。因此，这种生成均匀分布伪随机数的方法，对某些高精度初始化的应用提供了有益的尝试方向。

图 3 所示为 e 的负指数曲线。从图 3 可以看出，当实验次数 $N>5$ 时，参数效果提升已不明显；而从运算开销考虑，选取 $N=3$ 已足够满足日常对均匀分布伪随机数的精度需求。

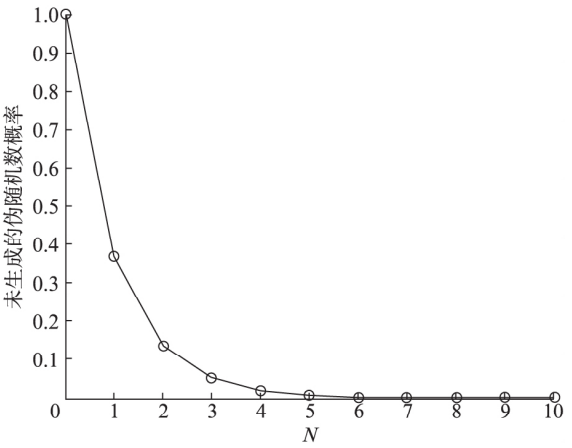


图 3　e 的负指数曲线

5　结　语

通过本文的讨论，可以看出使用伪随机数生成器逼近真实的随机数是目前各类研究中较为可行的一种方法。而使用本文中的评价方法，对所构建的伪随机数生成算法进行评价，可以有效地指导实际应用，并应用到各种群智能算法中，如粒子群算法[16]等。同时针对特定问题，也可以对已有的伪随机数算法进行简单的组合改进，进行评价和改进指导，以更好的解决问题。

（下转第 62 页）

semantic scale[J].Creativity Research Journal,1989,2(4):267-278.

[10] REDMOND M R, MUMFORD M D, TEACH R. Putting creativity to work:effects of leader behavior on subordinate creativity[J]. Organizational Behavior & Human Decision Processes, 1993, 55(1): 120-151.

[11] 陈信康,兰斓.基于消费者体验的产品创意维度构成及测量[J].管理评论,2012,24(6):66-73.

[12] 李碧珍.创意商品的价值构成与价值实现[J].当代经济研究,2007(9):27-30.

[13] HORN D,SALVENDY G.Product creativity:conceptual model measurement and characteristics[J]. Theoretical Issues in Ergonomics Science, 2006, 7(4):395-412.

[14] SWEENEY J C, SOUTAR G N. Consumer perceived value:the development of a multiple item scale [J].Journal of Retailing,2001,77(2):203-220.

[15] GOODSTEIN L D,BUTZ H E.Customer value:the linchpin of organizational change[J]. Organizational Dynamics,1998,27(1):21-34.

[16] GENTILE C,SPILLER N,NOCI G.How to sustain the customer experience an overview of experience components that co-create value with the customer [J]. European Management Journal, 2007, 25(5): 395-410.

(上接第 49 页)

参 考 文 献

[1] LU C, LI X, GAO L, et al. An effective multi-objective discrete virus optimization algorithm for flexible job-shop scheduling problem with controllable processing times[J]. Computers & Industrial Engineering,2017(104):156-174.

[2] YANG X S, DEB S, ZHAO Y X, et al. Swarm intelligence:past, present and future[J]. Soft Computing,2018,22(18):1-11.

[3] PINEDO M. Scheduling: theory, algorithms, and systems,5th[M].Berlin Heidelberg:Springer,2016: 1-7.

[4] YAO L,LIU Y B,ZHAO H B,et al.An improved UKPK-PSO algorithm inspired from block chain technology for flexible job shop scheduling problem [C]// 2019 Chinese Control Conference (CCC). Guangzhou:IEEE,2019:702-707.

[5] 郑秀莲,王万良.多产品批处理不确定生产调度的粒子群算法[J].工业控制计算机,2013,26(9):41-43.

[6] 方兴东,王俊秀.冯·诺伊曼:电子计算机之父[J].软件工程师,2007(12):60-63.

[7] GUOLO A, VARIN C. Random-effects meta-analysis:the number of studies matters[J]. Statistical Methods in Medical Research, 2015, 26(3): 1500-1518.

[8] 陈福玉,雷春怡,卢旻昊.空间中具有确定端点的不确定简单线及其向相关拓扑谓词的实现[J].电子质量,2018(4):24-28.

[9] 顾浩,丁豪杰.计算机结构与组成[M].北京:高等教育出版社,2010:1-15.

[10] BAO Y G,WANG S.Labeled yon Neumann architecture for software-defined cloud [J]. Journal of Computer Science and Technology, 2017, 32(2): 219-223.

[11] LAMINE S M, IBTISSEM B. A pseudo-random numbers generator based on a novel 3D chaotic map with an application to color image encryption[J]. Nonlinear Dynamics,2018,94(1):723-744.

[12] LU C J. Improved pseudorandom generators for combinatorial rectangles[J].Combinatorica,2002,22(3):417-433

[13] WOLPERT D H, MACREADY W G. Coevolutionary free lunches[J]. IEEE Transactions on Evolutionary Computation,2006,9(6):721-735.

[14] 师国栋,康绯,顾海文,等.随机性测试的研究与实现[J].计算机工程,2009,35(20):145-147,150.

[15] CLERC M. Particle swarm optimization[M]. Paris: John Wiley & Sons,2013:59-69.

[16] 赵睿智,丁云飞.基于粒子群优化极限学习机的风功率预测[J].上海电机学院学报,2019(4): 187-192.