

文章编号: 1000-5862(2013)06-0579-05

遗传算法在试卷生成中的应用

周莉¹, 王珏¹, 周勇²

(1. 华东交通大学软件学院, 江西 南昌 330013; 2. 江西师范大学计算机信息工程学院, 江西 南昌 330022)

摘要: 试卷自动生成是一个多目标的问题, 遗传算法可以搜索全局最优解, 将遗传算法用于自动组卷算法中, 并引入平衡算子和权重算子计算算法的概念, 有效地解决难以寻求最优解的问题. 实验结果表明: 该算法有效地避免了通过一些常规的方法所造成的弊端, 并且提高了搜索全局最优解的能力, 加快了收敛速度. 该方法不仅在教育领域有一定的实用性, 也具有潜在的应用价值.

关键词: 遗传算法; 组卷; 平衡算子; 权重算子

中图分类号: TP 315.69

文献标志码: A

0 引言

随着计算机技术和网络技术的发展, 计算机自动组卷技术在教育领域得到广泛应用. 一个好的生成试卷方法^[1-2]对于提高试卷生成的成功率起着十分重要的作用. 目前有一些普通的方法被广泛采用, 如随机选择和线性规划, 但得到的结果并不令人满意. 自动生成的问题是一个多目标的问题^[3], 并且还要知道所有的约束条件和目标不能达到的可能性. 为了克服这些困难, 人们通常使用解决多目标优化问题的启发式搜索技术.

对于那些预先指定的属性, 如每一项的分值、时间、类型、所有项的总数、每项的难度等, 都必须建立一个合理的数学模型来调整它们, 才可以得到满足用户需求的试卷. 众所周知, 遗传算法(GA)早已被确认为一个良好的自动参数提取方法, 而不再是费时费力的最优化方法. 它是一种流行的智能系统非经典数学方法. 因此, 本文利用GA提出了基于遗传算法的自动生成试卷的方法, 同时提出了一个使用平衡算子和权重算子计算算法的适应度技术, 并用一些实验结果^[4]来说明它的有效性.

1 试卷属性分析

任意一份试卷包含很多属性, 合理分布这些属性直接决定试卷的质量. 为了获得合理的分布, 给出一个实例, 分析试卷中的每个属性. 假设生成一份试

卷, 其中包含40个项目. 下面的数值列表的目标值为预期的试卷的属性列表: (i) 总分 $S = 100$; (ii) 时间 $T = 120 \text{ min}$; (iii) 题目数量 $N = 40$; (iv) 平均难度 $ND = 0.5$; (v) 题目类型数 $I = 8$.

1.1 题目类型和数量

在试卷中题目类型和题目数量通常是不变的属性. 因为一旦用户在试卷生成之前就指定它们, 该值将是不变的, 在这里将不再过多地描述它们.

在试卷中把 l 作为题目数量, 让 N_i 作为第 i 个题目类型中题目总数, 当用户有预指定时, 在整个试卷中让 N 表示项目数量的总和和目标值. 在一份令人满意的试卷中, 题目类型的数量达到的目标值 I , 并且所有题目类型 N_i 的题目总和等于目标值 N , 即

$$N_1 + N_2 + \cdots + N_I = \sum_{i=1}^I N_i = N.$$

1.2 分数

让 S 表示试卷的总分, S_i 表示第 i 类题型的分值, S_{ij} 表示第 i 类题型中的第 j 个题目的分数, 在目标试卷中每个题目的分数 S_{ij} 和它们的总分 S_i 都满足目标值 S .

1.3 难度

让 ND 表示试卷的平均难度, S 表示试卷的总分, \bar{S} 表示试卷的平均分值, S_{ij} 表示第 i 类题型中的第 j 个题目的分数, \bar{S}_{ij} 表示在第 i 类题型中的第 j 个题目的平均分, \bar{S}_{ij} 该项目的类型中的分值, D_{ij} 表示第 i 类题型中的第 j 个题目的难度.

收稿日期: 2013-09-19

基金项目: 国家自然科学基金(61165004)和华东交大校立科研基金(12RJ03, 13RJ02)资助项目.

作者简介: 周莉(1977-), 女, 江西南昌人, 讲师, 主要从事数据库方面的研究.

试卷的难度在约束属性中起着重要的作用. 在大多数情况下, 相同的试卷能得到不同的结果, 即分数也是截然不同的. 一般来说, 如果分数按正态分布^[5], 可以假设试卷具有良好的质量和满足要求. 同时它也表明试卷的平均难度是接近预期的, 在试卷中每个项目的难度将合理分配.

正态密度函数表示为 φ

$$\varphi_{\alpha, \sigma}(x) = e^{-\frac{(x-\alpha)^2}{2\sigma^2}} / (\sigma \sqrt{2\pi}), -\infty < x < \infty.$$

一般地, 给定 $x \in (\alpha - 3\sigma, \alpha + 3\sigma)$, 概率的精度约为 99.74%. 为了提高精度, 这里让 x 通常分布在 $[\alpha - 5\sigma, \alpha + 5\sigma]$. 若 $\alpha = 50$, 则 $x \in [0, 100]$. 当 $\alpha = 50, \sigma^2 = 100$ 时的正态密度函数如图 1 所示.

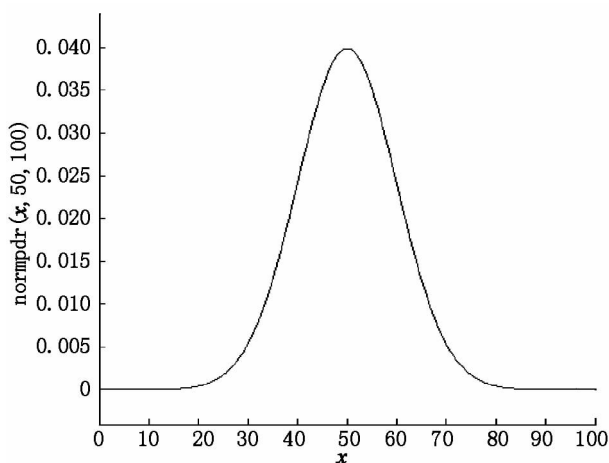


图1 平均值 $\alpha = 50$ 和方差 $\sigma^2 = 100$ 的正态密度函数

由图1可以看到正态密度函数是以50为轴对称的, 在50的位置有一个唯一的模式. 当预期的分值等于50, ND 的值由以下的等式计算得到: $ND = 1 - 50/100 = 0.5$.

由于试题难度从易到难逐步分级, 因此, 通过计算可以得到难度等级分布(见表1). 表1展示了当 $ND = 0.5$ 时, 整张试卷中相应的试题难度的分布.

表1 整张试卷中试题难度等级分布

难度	难度分布 /%	难度	难度分布 /%
0.1	0.003	0.6	34.130
0.2	0.132	0.7	13.595
0.3	2.140	0.8	2.140
0.4	13.595	0.9	0.132
0.5	34.130	1.0	0.003

如果已知一份试卷满分是100, 可以计算出每项试卷难度分布的分数. 例如, 当项目难度为0.5时, 相应的分数为 $100 \times 34.13\% = 34.13 \approx 34.1$.

显然, 当该试题的难度是0.1或1.0时, 计算出的

分值太小了, 它往往近似为 $0.100 \times 0.003 = 0.003 \approx 0$.

表2展示了根据上述原理, 不同的试题难度在目标试卷中的分值分布.

表2 一张优化的试卷中试题难度的分值分布

难度	分数	难度	分数
0.1	0	0.6	34.1
0.2	0.1	0.7	13.7
0.3	2.1	0.8	2.1
0.4	13.7	0.9	0.1
0.5	34.1	1.0	0

在本文中, 如果真正的平均难度的误差小于10%时, 也被视为是可接受的, 即

$$ND(1 - 10\%) < \left(\sum_{i=1}^I \sum_{j=1}^{N_i} S_{ij} D_{ij} \right) / S < ND(1 + 10\%).$$

1.4 考试时间

设 T 为完成整张试卷的总时间, 设 T_i 为第 i 类题型中所有试题的时间总和, T_{ij} 为第 i 类题型中第 j 项试题所消耗的时间. 于是有 $T_{i1} + T_{i2} + \cdots + T_{iN_i} = \sum_{j=1}^{N_i} T_{ij} = T_i$ 和 $T_1 + T_2 + \cdots + T_I = \sum_{i=1}^I T_i = T$. 因此, 若所有试题的时间总和的误差小于10 min, 则该试

卷被认为合格, 即 $T - 10 < \sum_{i=1}^I \sum_{j=1}^{N_i} T_{ij} < T + 10$.

对试题的合理定义, 在试题的时间和难度之间存在一定的相关性. 如果题型是相同的, 试题越难, 答题花费的时间越长, 反之亦然. 对于那些难度低, 又需要较多时间回答的试题, 就没有任何测验的价值, 必须将其筛选出来, 以避免降低试卷的质量. 至于所消耗的时间上的正态分布的问题, 可采用编程的方式, 它超出了本文的讨论, 为了简化讨论, 在本文中将其不认为是 GA 的一个因素.

2 遗传算法

遗传算法是达尔文的进化理论和孟德尔的遗传学的结合, 已成功地应用于解决很多领域的搜索优化问题^[6]. 首先, 随机产生个体的初始种群, 而每个个体用二进制编码或直接编码表示. 然后计算出适应度, 目的是评估样本集的分类精度. 接下来, 重复的一系列操作, 包括排序、选择、交叉和变异, 直到每个种群中每个个体的适应度都满足预先指定的适应度值, 它实现了进化的原理, 这就是“适者生存”.

与其它算法不同,GA 有其自身的优势:(i) 为了优化种群的多点搜索增加了得到全局最优解的概率,这种方法与单点搜索不同,可以避免收敛至局部最优解.(ii) 如果没有其它辅助信息,适应度是判断选择的种群是否为最优的一个关键的决定因素.此外,适应度函数可以任意定义,无论是连续的或可微的,显式函数或隐式函数.(iii) 它属于启发式搜索,因此搜索效率是非常高的.(iv) 它被广泛用于解决大规模并行计算的领域、智能网络和复杂大型系统中的优化问题.

3 一个基于遗传算法的试卷生成算法

在本文中设计了一个基于遗传算法的组卷算法来处理试卷生成中寻求最优解问题,如图 2 所示.

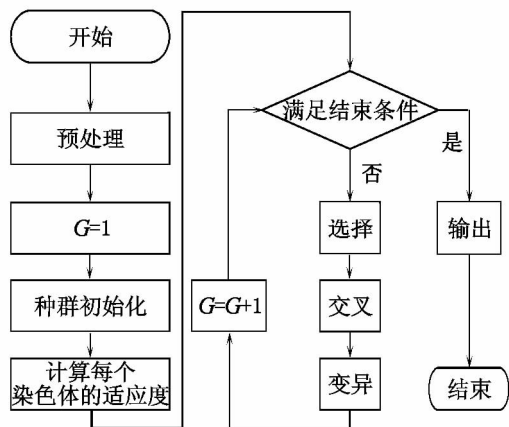


图 2 试卷生成算法流程图

3.1 预处理

为了提高搜索效率,先进行预处理,再执行遗传算法.它包括试题库的备份,并通过删除与用户的需求无关的试题对当前库进行筛选.此时,所有的新试题库中的试题都在用户预先定义的可选范围中.因此,优化试题库缩小了搜索空间,提高了效率.在文献[7]的实例,因为选择了所有的题型,不需要删除任何题型.但是有必要删除难度为 0.1 和 1.0 的试题,因为在目标测验中它们的分值为 0.

3.2 种群初始化

首先,在试题库中随机选择 n 项试题作为初始种群,其中 $n \geq 1$.令 Q_1 为第 1 代 G_1 的 1 个矩阵,其定义为 $Q_1 = [Q_{11} \ Q_{12} \ \cdots \ Q_{1n}]^T$, Q_{11}, Q_{12}, \cdots 是染色体的二进制串.

令 Q_{1i} 为其中一个染色体,定义为 1 个 13 比特的染色体.

(i) 题型表示为 3 比特序列.在本文中有 8 个

选项,如表 3 所示.

(ii) 分值表示为 4 比特,为了简化问题此处只有 16 个选项.

(iii) 由于剔除了试题难度为 0.1 或 1.0 的项,难度表示为 3 比特的序列,8 个选项如表 4 所示.

表 3 题型选择

选项	题型	选项	题型
000	I	100	V
001	II	101	VI
010	III	110	VII
011	IV	111	VIII

(iv) 时间表示为 3 比特序列,为了简化问题此处仅有 8 个选项.

表 4 难度选择

选项	难度	选项	难度
000	I (0.2)	100	V (0.6)
001	II (0.3)	101	VI (0.7)
010	III (0.4)	110	VII (0.8)
011	IV (0.5)	111	VIII (0.9)

3.3 适应度函数

考虑到目标测验是一张理想的试卷,其中平均难度和试题的数量都达到用户预期的目标值,这里介绍平衡算子 E 和权重算子 w . $E_{ij} = (D_{ij} - ND) S_{ij}$, $E = \sum \sum E_{ij}$,其中 ND 表示在试卷的平均难度的目标值, S_{ij} 表示第 i 类题型中的第 j 个题目的分数, D_{ij} 表示第 i 类题型中的第 j 个题目的难度,给定 $D_{ij} = (0.2 \ 0.3 \ 0.4 \ 0.5 \ 0.6 \ 0.7 \ 0.8 \ 0.9)$.

可以由 $de = E / \left(\sum_{i=1}^i \sum_{j=1}^j S_{ij} \right) = \sum_{i=1}^i \sum_{j=1}^j (D_{ij} - ND) \cdot S_{ij} / \left(\sum_{i=1}^i \sum_{j=1}^j S_{ij} \right)$ 计算出当前的偏差 de .

如果平衡算子 E 通过计算为 0,它表明目前的平均难度等于目标值 ND ,即 $de = ND$.在这种情况下,认为目前的试卷平均难度处于平衡状态.当前的状态是合理的.

通过计算如果是负数,它说明目前的平均难度是小于目标值 ND ,这就是说 $de < ND$.在这种情况下,认为目前试卷的平均难度低于平衡的. E 代表需要达到平衡的平衡量,反之亦然.当试题的难度是 0.4,其目标分值分布是 13.7. 如果从种群中得到的试题难度的分值总和是 12,它的权重算子 w_4 根据 $w_4 = 12 / 18.7 \approx 0.876$ 算出.然后,计算 w 为 $w = \sum_{k=1}^8 w_k$.

一方面, de 越小,试卷的平均难度近似地增大.另一方面, w 越大,目标测验获得的准确度更大.因

此,可以定义适应度函数为 $f = w(1 - |del|)$.

3.4 选择

选择是 GA 的过滤器,它决定了哪些解成为双亲,哪些解被丢弃.这里使用轮盘赌选择^[8]来创建新一代.

3.5 交叉

交叉使得 GA 创建与其双亲不同的子串,但仍然含有相同的遗传信息^[9].在本文中,只交换 2 个二进制串,并基于相同的题型使用部分匹配交叉^[10-11](PMX)产生 2 个新的二进制字符串.比如,首先,随机选择 2 个双亲串进行观察.交叉前,

$$P_1 = \begin{bmatrix} 010 & 1010 & 011 & 101 \\ 110 & 0111 & 101 & 011 \\ \vdots & & \vdots & \\ 011 & 1110 & 000 & 010 \\ 001 & 0001 & 111 & 110 \end{bmatrix},$$

$$P_2 = \begin{bmatrix} 100 & 0001 & 010 & 000 \\ 110 & 1111 & 111 & 001 \\ \vdots & & \vdots & \\ 000 & 0100 & 111 & 101 \\ 110 & 1111 & 000 & 011 \end{bmatrix},$$

交叉后,创建了 2 个新的后代:

$$Q_1 = \begin{bmatrix} 010 & 0001 & 010 & 101 \\ 110 & 1111 & 111 & 011 \\ \vdots & & \vdots & \\ 011 & 0100 & 111 & 010 \\ 001 & 1111 & 000 & 110 \end{bmatrix},$$

$$Q_2 = \begin{bmatrix} 100 & 1010 & 011 & 000 \\ 110 & 0111 & 101 & 001 \\ \vdots & & \vdots & \\ 000 & 1110 & 000 & 101 \\ 110 & 0001 & 111 & 011 \end{bmatrix}.$$

3.6 变异

变异将新的信息引入种群,同时增加生殖过程中的噪音.如种群中的每一比特有很小的概率被从 1 翻转到 0,反之亦然.变异率是 1 个可变的 GA 参数.如果它被设置为 0.003,选择了 40 个项目,就由 $40 \times 13 \times 0.004 = 2.080$ 计算变异比特,从而可知变异发生 2 比特.

3.7 循环和相交

新的种群 G_2 被创建,但其仍然需要被发送到评估程序,将每个个体解码成参数值,使用这些参数运行 1 次模拟,而且种群成员被分配 1 个适应度值.一旦评估完成,种群准备进行另一次循环的 GA. GA 可以监控运行过程中最佳一代的适应度,当达到一

定的适应质量或连续几代都没有更好的解时将停止循环.终止条件为:(i) 整体适应度是在期望的区域内,难度的分布是相当合理的,即整张纸卷相应的试题难度大致服从正态分布,并且它的误差范围是 0.05 ~ 0.50. (ii) 代数到达预先指定的最大值. (iii) 题库中的所有试题的总数不足,即没有更多的试题可以被选择,因此该算法不得不停止并以失败告终.

4 实验和结果

4.1 实例

为了证明所提出的二进制编码 GA 的有效性,通过一些实验来获得给定的交叉算子和变异算子的试卷生成问题的最佳解决方案.

在实验中,指定的交叉算子是 0.400,变异算子是 0.004.实验结果见表 6.

表 6 实验结果

实验	人数	花费时间
1	15	287.301
2	10	99.663
3	25	597.798
4	10	53.216

图 2 给出了在一个实验中,从试卷中得到的相应的试题难度的分布情况.

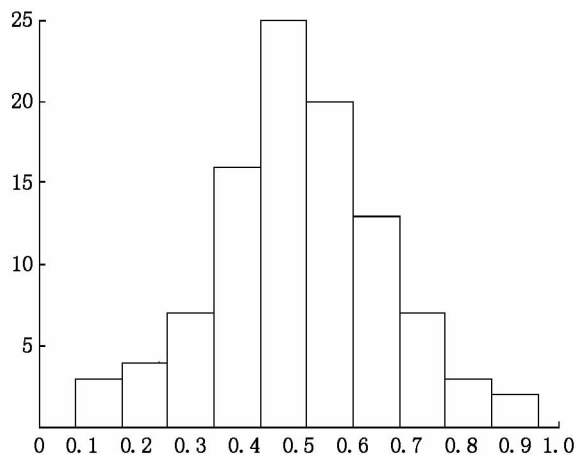


图 2 难度分布示意图

4.2 结果和比较

观察上述实验中的数据,可以发现初始种群一般是不好的,因为它是任意选择的.但是,通过一系列的进化,因为其发展以目标为导向,它们的后代变得越来越好.为了看得清楚,也做了常规的随机选择和 GA 的有效性的比较.表 7 给出的 2 种方法之间的比较.

表 7 随机选择法和遗传算法的误差范围

随机选择		遗传算法	
最大	最小	最大	最小
0.424	0.213	0.072	0.017

由表 7 可知 GA 是在性能上优于随机选择的.

5 结论

作为一种能解决复杂性和全局优化问题的良好的仿生搜索算法,GA 被用于智能组卷中的试题选择. 具有较高的成功率和较低的误差精度. 为了将问题描述清楚,简化问题,忽略了试题上所耗费的时间,但实际上,试题的时间也是一个重要因素,却往往都被忽视. 试题的时间安排是否合理影响回答试题的速度,进而影响整份试卷的分数. 对于不同的试题类型,试题的时间也是不同,因此,在这种情况下无法比较,但是如果试题属于同一类型,则可以比较其时间. 在本文中,假设试题的指定时间围绕基本时间上下波动,最好其时间分布服从以基本时间为对称轴的正态分布. 将来的工作将侧重于研究如何在算法中融入一种想法,即在试题的类型相同的情况下,如何使得其时间服从正态分布. 同时,将定量地指出提高精度和改善算法的难度.

6 参考文献

[1] Itchell M. An introduction to genetic algorithms [M].

Cambridge: MIT Press,1996: 15-17.

[2] 王月敏. 基于遗传算法的智能组卷系统的研究 [J]. 云南民族大学学报: 自然科学版, 2009(4): 173-176.

[3] Sahoo L, Bhunia A K. Genetic algorithm based multi-objective reliability optimization in interval environment [J]. Computers & Industrial Engineering, 2012, 62(1): 133-139.

[4] Takahashi T, Nakai H. An automatic modeling system of the reaction mechanisms for chemical vapor deposition processes using real-coded genetic algorithms [J]. Journal of Nanoscience and Nanotechnology, 2011, 11(9): 219-224.

[5] Linhart J M. Algorithm 885: computing the logarithm of the normal distribution [J]. ACM Transactions on Mathematical Software, 2009, 25(3): 102-109.

[6] Machmudah M, Parman S. UAV bezier curve maneuver planning using genetic algorithm [C]. New York: ACM, 2010.

[7] Dyer J D, Hartfield R J. Aerospace design optimization using a steady state real-coded genetic algorithm [J]. Applied Mathematics and Computation, 2012, 218(69): 43-45.

[8] 王小平, 曹立平. 遗传算法理论应用与软件实现 [M]. 西安: 西安交通大学出版社, 2002.

[9] 何春华, 胡迎春. 遗传算法在自动组卷中的应用 [J]. 广西工学院学报, 2006, 17(4): 60-63.

[10] Acharjee P. Identification of maximum loadability limit and weak buses using security constraint genetic algorithm [J]. International Journal of Electrical Power and Energy Systems, 2012, 36(1): 73-75.

Application of Test Paper Generation Based on Genetic Algorithm

ZHOU Li¹, WANG Jue¹, ZHOU Yong²

(1. School of Software, East China Jiaotong University, Nanchang Jiangxi 330013, China;

2. College of Computer and Information Engineering, Jiangxi Normal University, Nanchang Jiangxi 330022, China)

Abstract: Considering the problem on generating test papers is multi-objective and the genetic algorithm (GA) can search the globally optimal solution, a GA-based algorithm for test paper generation automatically is presented and effectively solves the problem that it is hard to find the optimal solution by introducing the concepts of the equilibrium operator and the weight operator. The experiments show that this algorithm effectively avoids the disadvantages caused by some conventional methods and moreover it improves the ability of searching a globally optimal solution and increases the convergent speed. This method not only is effective for the kind of practical problems in education field but also has potential applications in several problems in engineering.

Key words: Genetic Algorithm; test paper generation; equilibrium operator; weight operator

(责任编辑: 冉小晓)