题库自动组卷算法的研究

赖 清
(重庆大学 重庆 400044)

摘 要 根据教育考试理论与统计学,分析了自动组卷目标要求,建立了智能组卷系统的数学模型,并对几种常用的组卷算法进行了设计与比较。

关键词 组卷目标,数学模型,组卷算法

1 引言

由于远程教育和网络教育的发展,题库系统的研制一直是一个非常活跃的课题。在题库系统的研制过程中,各功能模块的划分与封装、实现方法等,都有其相似性,一个高效、实用的题库系统,其核心是智能化的自动组卷。自动组卷是各种题库实现系统自动和半自动化操作的关键,而如何保证在考试内容和考试要求相同的情况下,在短时间内得出大量差别较大的等效试卷充分满足用户需要,则是实现中的一个难点。

组卷系统首先需要调用相应的组卷策略,设计出符合用户要求和一定约束条件的试卷模式,然后再按试卷模式选取试题组成试卷。组卷系统具有公正客观、快速高效的优点可以提高考试的质量和成绩的可信度,能够满足网络教育及成绩的可信度,能够满足网络教育及学校教育的需要。

2 自动组卷问题

自动组卷即在系统题库中,根据用户输入的组卷要求,搜索试题库中特征参数相匹配的试题,生成满足要求的试卷。

2.1 与组卷有关的试题参数

在智能教学系统的研究中,一个非常重要的课题是怎样在已生成的题库中自动生成满足教学和教师要求的试卷。通常,全部试题按内容分为若干章节、题目类型类和多个细目,每一道试题又包含多个属性,其中与组卷有关的属性有如下12项:

(1)章节:试题内容所属章节;

(2)试题编号:试题编号具有与每一道试题一一对应的性质,它不参与在自动组卷中的运算,仅作为参与运算的数据结构的不变分量,用于指示运算结果具体记录;

(3)试题类型:一般有选择题、填充题、计算题、证明题、改错题、问答题、判断题等;为了在生成试卷时,能够快速选取指定类型的试题,和减少冗余度,在建库时可为每种题型建立一个库文件;

(4)试题难度系数:在试卷命题过程中,针对不同的考试对象,不同阶段的考试,命题难度也不同,所以应在数据库中增加难度系数。Ndxs=1-(平均分/该题满分),有容易(得分率:0.8~1.0)、中等(得分率:0.6~0.8)、偏难(得分率:0.3~0.6)、难(得分率:<0.3)四个级别;

(5)试题的内容:不参与组卷运算的过程;

(6)层次:教学内容精熟程度,可分为识记、理解、应用、综合等;

(7)区分度:试题对考生的知识能力水平鉴别和区分程度的指标,分3级:知识,运用,灵活应用;

(8)已出题次数:用于表明已出题次数,以此决定此题再出的概率,这一结构对出题的影响是:结构中的值越大,再出的概率越小,提高命题质量;

(9)试题内容的相关性[1]:在试卷命题过程中避免相同内容的试题太多,从而无法对学生所学的知识进行全面的考核,因此可用平均相关系数和相关性分布η两个指标来考核。一份试卷的平均相关系数应在3~7之间;

(10)考查点:为了适应不同教材的需要,可按大纲要求将某门课的内容分为不同的考查点,选题时以考查点为依据,同一试卷中不能有考查点重复的试题,它是和相关性密切相关的结构;

(11)题分:试题的分数;

(12)估时:完成该题所需时间的估计值。

2.2 数学模型

组卷中决定一道试题,就是决定它的上述12个属性,也就是说决定一个12维的向量($\alpha_1,\alpha_2,\alpha_3,\cdots,\alpha_{12}$),决定一份n道试题,实际上就是决定一个n×12的矩阵:

$$S=\begin{pmatrix}\alpha_{1,1} & \alpha_{1,2} & \cdots & \alpha_{1,12}\\ \alpha_{2,1} & \alpha_{2,2} & \cdots & \alpha_{2,12}\\ \vdots & \vdots & \vdots & \vdots\\ \alpha_{n,1} & \alpha_{n,2} & \cdots & \alpha_{n,12}\end{pmatrix}$$

这就是一个问题求解中的目标状态矩阵,从上

面指出的组卷的功能要求可以看出，目标状态矩阵应满足如下相应的约束条件：

(1)试卷的总分$=\sum_{i=1}^{n}\alpha_{i,11}$由用户给定，即试卷分数约束。

(2)试卷的平均难度$=\sum_{i=1}^{n}\alpha_{i,11}\alpha_{i,4}/$总分，由用户给定，即试卷难度约束。

(3)考试时间$=\sum_{i=1}^{n}\alpha_{i,12}$由用户给定，即试卷总时间的约束。

同理，教学要求、各种题型比例等约束条件，由用户在组卷时给定。但根据经验，指标过多对组卷问题增加难度降低效率，因此，在组卷时可以根据用户组卷要求给出相应的约束条件。可以看出，组卷问题的实际是求多约束条件的最优解，而且满足条件的最优解不是唯一的。

3 自动化组卷算法的设计

3.1 随机选取法

随机选取法根据状态空间的控制指标，随机的抽取一道试题，此过程不断重复，直到组卷完毕，或已无法从题库中抽取满足条件的试题为止。算法为[2]：

1)建立2个数组Z(S)，U(X)。Z(S)的值为某种状态S的试题在题库中的总题量，如选择题、第5章、难度为3的试题总数。U(X)为用户要求的该状态的试题数目，U(X)对应的全部状态的集合构成线形表LIST；

2)如U(X)>Z(S)，则转向5)，否则产生随机整数N1，N1 = int(rand(-1)×Z(S))，其值小于Z(S)。读取N1记录，并对该记录作读取标志，抽取下一道题目时，有读取标志的记录不再有效；

3)若U(X)≠0，则U(X)= U(X)-1，重复2)；

4)若LIST表未满，则转向1)，否则组卷成功；

5)算法结束。

该算法结构简单，对于单道题的抽取运行速度较快，但是对于整个组卷过程来说组卷成功率低，即使组卷成功，花费时间也令人难以容忍。

3.2 回溯试探法

回溯试探法是将随机选取法产生的每一状态都记录下来，当搜索失败时释放上次记录的状态类型，然后再依据一定的规律(正是这种规律破坏了选取试题的随机性)变换一种新的状态类型进行试探，通过不断地回溯试探直到试卷生成完毕或退回出发点为止。算法为[3]：

1)建立2个数组Z(S)，U(X)，线形表LIST，含义同随机选取法的算法；

2)如U(X)>Z(S)，则①按一定策略减少状态S的U(X)的值；②增加与之相接近的另一状态U(X)的值，并修改相应的LIST值，直到U(X)<Z(S)为止。否则，产生随机整数N1，N1=int(rand(-1)×Z(S))，其值小于Z(S)。读取N1记录，并对该记录作读取标志，抽取下一道题目时，有读取标志的记录不再有效；

3)U(X)≠0，则U(X)= U(X)-1，重复2)；

4)若LIST表未满，则转向1)，否则组卷成功；

5)算法结束。

这种有条件的深度优先法，对于状态类型和出题量都较小的题库系统而言，组卷成功率较好，但是在实际到一个应用时发现这种算法对内存的占用量大，程序结构相对比较复杂，而且选取试题缺乏随机性，组卷时间长，后两点是用户无法接受的，因此它不是一个很好的用来自动组卷的算法。

3.3 随机化启发式搜索法

随机化启发式搜索法是对前2种算法的改进。在搜索的前几步中采用随机选取法，当搜索进入死结点时，采用下面的方法作启发后再试：首先将造成死结点的状态类型记录下来，然后回溯走过的路径，将与该状态有关(指某分量相同)的元素全部释放，将剩下的无关元素重新构成一路径，然后根据启发函数的最小值确定下一个结点，启发函数定义为新元素状态类型与记录死结点元素状态类型分量相同的个数。算法为[4]：

1)建立2个数组Z(S)，U(X)，线形表LIST，含义同前。再建立表DEAD用于记录历次死结点的状态，启发函数H(S)表示状态类型S与DEAD表中各状态类型的最大相似度，即最大相同分量数；

2)若DEAD表为空，则随机选取符合用户要求的状态类型S；否则，在所有未试过的符合用户要求的状态类型P中随机选取H(S)具有最小值的状态类型S；

3)若U(X)≠0，则① U(X)=U(X)-1；②将S加入LIST表中，否则，若尚有符合用户要求的其他状态未试，则重复2)；否则，(a)从LIST表中除去与S有相同分量的状态类型；(b)将S加入DEAD表中；(c)若DEAD表中有相同的状态类型，则重置U(X)数组，本次组卷尝试失败，转向5)，否则转向2)；

4)若LIST表未满，则转向1)，否则组卷成功；

5)算法结束。

若用户的组卷要求合理，则该算法组卷成功率高，且用户等待时间短，但该算法增加了程序设计的复杂度。

3.4 遗传算法

遗传算法(Genetic Algorithm,GA)是近年来迅速发展起来的一种全新的随机搜索与优化算法,其基本思想是基于 Darwin 的进化论和 Mendel 的遗传学说。该算法由密执安大学教授 Hol2land 及其学生于 1975 年创建[5]。

遗传算法是一种模拟自然界生物进化过程的计算模型,遗传算法能解决随机算法的盲目随机性,并能从群体中选择更满足条件的个体,具有很强的智能性,同时它能根据不同的环境产生不同的后代,具有动态性,自适应性,从而能满足题库不断变化的要求,由于试题库容量大,覆盖面广,因此选题计算量大,利用遗传算法的内在并行性,可以有效的解决计算量大的问题。

与传统搜索算法不同,遗传算法从一组随机产生的初始解,称为群体,开始搜索过程。群体中的每个个体是问题的一个解,称为染色体。这些染色体在后续迭代中不断进化,称为遗传。遗传算法主要通过交叉、变异、选择运算实现。交叉或变异运算生成下一代染色体,称为后代。染色体的好坏用适应度来衡量。根据适应度的大小从上一代和后代中选择一定数量的个体,作为下一代群体,再继续进化,这样经过若干代之后,算法收敛于最好的染色体,它很可能就是问题的最优解或次优解。

遗传算法的实现[6]如下:

1)生成初始群体;

2)计算适应值;

3)根据适应值,选择染色体进行复制;

4)进行交叉、变异等操作生成新一代群体;

5)计算适应值;

6)若改变交叉、变异的概率条件成立,则改变;

7)如果最好个体满足所有上述约束条件,则输出当前最好个体,否则转向 3)。

遗传算法主要通过交叉算子繁衍后代,当交叉算子所作用的两个个体相同时,不能产生有意义的新的个体,因此要求初始种群体具有广泛多样性。当群体进化到其中的各个个体均相同时,交叉算子无效,此时仅靠变异算子产生新的个体,遗传迭代难以进行下去,即发生所谓"早熟收敛"现象。

结束语 一个自动组卷系统的性能主要取决于组卷算法,一个好的组卷算法既要保证组卷的成功率,又要保证数据运算的时间效率。在传统的组卷算法中(诸如随机抽取法,回溯法),组卷成功率较低,时间和空间开销都比较大,适合于小型题库系统。将遗传算法应用于组卷中,使组卷的成功率和收敛速度都得到明显提高,适合于较大型题库系统。由于求解精度和收敛速度是相互矛盾的,要使组卷的误差精度和收敛速度进一步得到改进,还需要做出更深入的研究。

参 考 文 献

1 毛秉毅. 基于遗传算法的智能组卷系统数据库结构的研究[J]. 计算机工程与应用,2003(6):2311

2 李小勇,刘开生,王瑛. 题库管理系统的设计与实现[A]. 计算机应用研究[C]. 北京:北京工业大学出版社,2000. 245~246

3 李小勇,王瑛. 题库管理系统中的自动化组卷算法[J]. 西北师范大学学报(自然科学版),2002

4 徐娟芬,袁小东. Pascal 题库系统的设计与实现[J]. 计算机应用,1998,18(6):16~19

5 Holland J H. Adaptation in Natural and Artificial Systems. Ann Arbor:University of Michigan press,1975

6 FEN G Xudong, CHEN G Fan. Usage and realization in genetic algorithm program ring [J]. Development in Microcomputers, 1997, 6: 426

(上接第 204 页)

```
      =new PWaveCurveGraphView;
    break;
    case SWAVE
      SwaveCurveGraphView NewActiveView
      =new SWaveCurveGraphView;
    break;
    ......
  }
  SetActiveView(NewActiveView);
  NewActiveView. ShowWindow();
}
```

结束语 软件系统日益复杂,面向对象应用框架作为面向对象系统获得最大复用的方式,变得越来越普遍和重要,因而有着广阔的发展前景。目前,已经有相当数量的成熟框架可供选择。根据特定的面向对象系统的特点,使用已有框架进行系统开发是一种普遍使用和行之有效的开发策略。成熟框架通常使用多种设计模式。了解框架的运行机制,了解这种机制所体现的设计模式的思想,并将设计模式的思想运用到对框架的使用和扩展上,有助于开发出更为健壮和更易扩展的面向对象系统,且有助于归纳特定领域设计经验,提取特定领域的设计模式,在已有框架基础上形成自己的领域框架。

参 考 文 献

1 Johnson R E. Framework=Components+Patterns [J]. Communications of the ACM,1997,40(10):39~42

2 Alexander C, Ishikawa S, Silverstein M. A Pattern Language: Towns, Buildings, Construction [M]. New York: Oxford University Press,1997

3 Fayad M, Schmidt D. Object-oriented Application Framework [J]. Communications of ACM,1997,40(10):32~38

4 Gamma E, Helm R, Johnson R, et al. Design Patterns: Elements of Reusable Object-Oriented Software [M]. Reading Massachusetts: Addison-Wesley Publishing Company,1995

5 李英军,吕建,王宏琳. 面向对象应用框架在油气勘探领域的应用研究[J]. 软件学报,1999,10(4):349~354

6 何 昭,李传湘,崔巍. 基于面向对象框架的软件开发方法[J]. 计算机工程,2002,28(4):5~7