DOI: 10.13657/j.cnki.gxkxyxb.1998.01.011

第 14卷第 1期 广西科学院学报 Vol. 14, No. 1
1998年 2月 Journal of Guangxi Academy of Sciences February 1998

普通高校课务管理排考系统的算法设计

Algorithmic Device for the Examination Arrangement System for Colleges and Universities

徐尚进

Xu Shangjin

(广西大学数学与信息科学系 南宁 530004)

(Dept. of Math. and Info. Sci., Guangxi Univ., Nanning, 530004)

摘要 针对普通高校排考的复杂性,定义了相关数据集,基本解决了排考过程中的多因素问题,使排考结果结合了计算机的技术特点和传统习惯。其中算法 (I) 已在广西大学计算机排考中正式采用,深受有关部门的欢迎。

关键词 算法 相关数据 排考试

Abstract Two basic algorithms to arrange examination by computer for colleges and universities were given. It was algorithm (I), especially, that allows full play to action to computer in practice so that efficiently to arrange the examination was promoted.

Key words algorithms, correlative data, arrangement of examination

中图法分类号 TP 311.12 G 743.4

普通高校的期末排考工作任务艰巨复杂,然而这一繁重的数据处理工作长期以来主要靠手工完成,使排考效果既不科学也耗费了大量人力。这与当今计算机技术的飞速发展和广泛运用极不相称。

事实上也曾有人在研究和开发计算机排考系统,但由于未能很好解决排考中诸多相关因素的算法问题,因此这些系统往往不能使用。这也是排考工作目前还停滞在手工处理阶段的一个最主要原因。

本文从建立排考相关数据入手,基本解决了排考过程中的多因素问题。叙述的是经简化和抽象后的两种排考基本算法。

1 问题和要求

每次期末排考,通常要考虑诸多因素,下边所提出的问题和要求对大多数普通高校都具

1997-10-22收稿。

有代表性:

(1) 考试时间要限制在一个时间段 (如两周) 内, 而且中间还可能有不能排考的日期 (如星期六、星期日等);

(2) 各班级考试课程的门数可以不等, 但每个班级各门课的考试日期间隔应尽可能均匀 (如 2周内考 3门课的班级应间隔 4 d, 而考 5门课的班级应间隔 2 d);

(3)各班级最后一门考试课程应尽可能安排在最后 1 d或倒数第 2天考,以免这些班的学生会妨碍他人复习;

(4) 考同一门课程 (称统考课程) 的所有班级必须安排在同一时间考;

(5) 安排考试教室要以班为单位, 教室考位数大于或等于该班人数, 否则要分成 2个教室, 但最多只能分 3个教室排考;

(6) 每个教室要安排主考教师 1名、监考教师 1名。一般由任该课程的主讲教师担任主考, 辅导教师担任监考。上合班课程的教师或同一门统考课程上多个班的教师, 主讲教师应主考其中一个班,而其余班必须安排在同一栋楼 (以便于分试卷和处理考试中的问题),如果该课程还配有辅导教师, 则应安排他在其余的某 1个班任主考。

(7) 有特殊要求的课程要安排在特殊教室考试, 如制图考试应安排在制图教室, 听力考试应安排在语音教室等。

(8) 某些课程 (如大学英语) 的考试, 要另外分班 (如 A B班) 排考, 但考试日期应与其它考试日期同时考虑。

(9) 还有一些要求是必须满足的: 如每个班在同一时间内只能考一门课程, 每个教室在同一时间内只能排一个班考试, 主考、监考教师在同一时间内只能主考或监考一个班。

上述要考虑的诸多因素往往使计算机排考工作陷入困境。

2 算法设计

计算机排考算法设计要满足上述各项要求, 首先要处理好与排考有关的数据。

2.1 基本数据

用集合表示:

K = {考试课程全体}

统考课程作为一门课, 每门课程有参加考试的班数 (称统考班数) 和总人数等指标, 按统考班数、总人数排序;

M = {参加考试班集全体}

以自然班为单位, 每个班有考试课程及其门数和人数等指标, 按人数排序;

C = {可用于排考的教室全体}

有考试座位数和教室类等指标, 按考试座位数排序;

T = {日期区间}

日期必须是连续的, 以便处理每个班考试日期均匀间隔问题, 不能排考的日期 (如周六和周日) 也要包含在内 (比如星期五和下星期一应按间隔 2 d对待);

H = {任课教师全体}

由于任课教师要参与主考或监考, 所以排考时也应考虑。

担任监考等因素，使课程的排考日期和教室往往不能独立确定；另外某一日期的某一教室可能已被先排的班级占用，选择教室时也应考虑。由此看来，基本数据之间相关的因素很多，我们称之为相关数据。定义和处理好这些相关数据是排考算法的关键。

2.2 相关数据

用子集合表示

取 $k\in K$,记

$M(k)=$ {考课程 k的班级全体}$\subset M$;

取 $m\in M$,记

$K(m)=$ {班级 m所要考试的课程}$\subset K$;

$T(m)=$ {班级 m已占用的考试日期}$\subset T$;

$\overline{T(m)}=$ {班级 m可继续排考的考试日期}$\subset T$;

取 $t\in C$,记

$C(t)=$ {日期 t已排考的教室}$\subset C$;

$\overline{C(t)}=$ {日期 t可排考的教室}$\subset C$;

取 $m\in M, k\in K$,记

$H(m,k)=$ {班级 m上课程 k的任课教师}$\subset H$;

此外,每个班要定义一个日期间隔函数,以便解决问题 2

取 $m\in M$,记

$J(m)=$ {班级 m最理想的考试日期间隔天数};

一般 $J(m)=\dfrac{\text{考试天数}}{\text{班级 } m \text{ 考试门数}}$. (要求取整)

文中上划线均表示取补集

2.3 基本算法

2.3.1 算法(I)的设计

(1) 先对考试课程库按统考班数、总人数排序 (对特殊需要的课程可优先排序)。

记 $K(i)=$ {已排考的 i门课程},显然 $K(i)$单调上升,

$\overline{K(i)}=$ {第 $i+1$门课程及其以后的课程}.

特别,记 $K(0)=\Phi, \overline{K(0)}=K$.

(2) 从 $i=1$开始。

(3) 取 $k_i\in \overline{K(i-1)}$. (取序号靠前的)

考虑 $M(k_i)$,

求 $T(M(k_i))=\bigcup_{m\in M(k_i)} T(m)$. ($M(k_i)$已占用的日期)

记 $T_i=T\setminus T(M(k_i))$. (表示取差集)

(4) 若 $T_i=\Phi$,则课程 k_i排不下,参见本文后注。

(5) 取 $t_i\in T_i$, (注: t_i必须考虑 $J(m)$及 $H(m,k_i)$,并优先取序号最后的,即考试日期靠后。)

考虑集合 $M(k_i)$与$\overline{C(t_i)}$,

若 $|M(k_i)|>|\overline{C(t_i)}|$,说明教室数已不够,则

若 $|M(k_i)| \leqslant |\overline{C(t_i)}|$,则

建立集合 $M(k_i)$ 与 $\overline{C(t_i)}$ 之间的对应关系 (或映射 f),要求:

对任意 $m \in M(k_i)$,存在 $f(m) = c \in \overline{C(t_i)}$,且 $|m| \leqslant |c|$. (即班级 m 人数不超过教室 c 座位数。)

这时课程 k_i 排考完毕。

(6) 若 $K(i) = K$,则全部排考完毕。

(7) 取 $i = i + 1$, 返回 (3) 继续。

2.3.2 算法 (Ⅱ)的设计

(1) 记 $M(i) =$ {已排的 i 个班级},显然 $M(i)$ 单调上升。

$\overline{M(i)} =$ {排考 i 个班级后剩余的班级}.

特别,记

$M(0) = \mathrm{H}, \overline{M(0)} = M$.

(2) 记 $t(k) =$ {课程 k 已排定的日期}

初始化,即对所有的 $k \in K$,令 $t(k) = 0$. (表示该课程尚未排定日期。)

(3) 从 $i = 1$ 开始。

(4) 取 $m_i \in \overline{M(i-1)}$. (取序号最前的)

考虑 $K(m_i)$,不妨记 $K(m_i) = \{k_1, \cdots, k_n\}$. (即班级 m_i 考 n 门课程)

(5) 从 $j = 1$ 开始。

(6) 取 $k_j \in K(m_i)$.

(7) 若 $t(k_j) = 0$, (即课程 k_j 未排) 则跳到 (9) 继续。

(8) 否则课程 k_j 已排在 $t(k_j)$,可取 $t_i = t(k_j)$,并跳到 (10) 继续。

(9) 记 $T_i = T \setminus T(m_i)$. (表示取差集)

取 $t_i = T_i$. (注: t_i 必须考虑 $J(m)$ 及 $H(m, k_j)$,并优先取序号最后的,即考试日期靠后)

(10) 考虑集合 $M(k_j)$ 与 $\overline{C(t_i)}$,建立集合 $M(k_j)$ 与 $\overline{C(t_i)}$ 之间的对应关系 (或映射 f),要求:

对任意 $m \in M(k_j)$,存在 $f(m) = c \in \overline{C(t_i)}$,且 $|m| \leqslant |c|$

这时课程 k_j 排考完毕。

(11) 若 $j < n$,则 $j = j + 1$,返回 (6) 继续。

(12) $M(i) = M$,则全部排考完毕。

(13) 否则,取 $i = i + 1$, 返回 (4) 继续。

算法 (Ⅰ) 以课程库 K为主进行排考,排完 K 中每一门课程即可;算法 (Ⅱ) 以班级库 M 为主进行排考,排完 M 中每一个班级即可。

最后要说明的是,随着排考的继续,可用的教室和日期将越来越少,甚至排不下去。这时候有两种可能: 一是教室数量本来就不够用; 二是通过一些特殊的处理,可以继续排下去。

如算法 (Ⅰ) 之 (4): 这时课程 k_i 已排不下,但如果教室数量从总体来说是够的,则仍可对前边已排课程进行适当调整再安排课程 k_i (方法在此从略); 如算法 (Ⅱ) 之 (10): 若 $|M(k_j)| > |\overline{C(t_i)}|$,说明教室数已不够,以后的处理比较麻烦。在实际使用中,采用算法 (Ⅰ) 比较简洁。