排课问题

排课问题其本质是将课程，教师，学生在合适的时间段分配到合适的教室中，即时间表问题（Timetable Problem）。

1.排课问题的数学模型：

1.1.时间表问题

时间表问题是一类多元受限的资源调度组合优化问题。

1.1.1定义：

1.时间集（Time Set)：事件发生时间构成的集合，具有有序性，唯一性，记为：

$T = \{t_1, t_2, t_3, \ldots, t_m\}$

2.空间集（Position Set)：事件发生空间构成的集合，具有无序性，唯一性，记为：

$P = \{p_1, p_2, p_3, \ldots, p_n\}$

3.资源集（Resource Set)：时间集T与空间集P的笛卡尔积成为资源集R，具有无序性，唯一性，其元素是一个由时间元素t_i与空间元素p_i组成的二元组，记为：

$R = \{r_1, r_2, \ldots, r_{m*n}\}$

$r_x =< t_i, p_j >, 1 \le x \le m*n, 1 \le i \le m, 1 \le j \le n$

4.事件集（Event Set）：只需要耗费一个资源元素就可以发生的某一过程或者行为构成的集合，记为：

$E = \{e_1, e_2, e_3, \ldots, e_k\}$

1.1.2描述：

时间表问题可以抽取出主要制约因素，如时间，地点以及在时间，地点上发生的事件，并将这些因素合理地划分成若干个集合。

按照实际时间表问题必须遵循的规则，事件间的偏序关系以及提高地点使用率等原则调度各个事件发生的时间与地点，可以得到诸如图1所示的事件发生图。

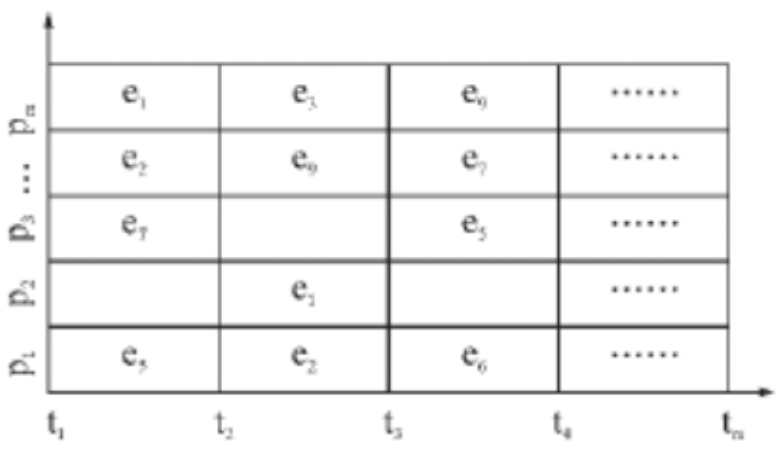

图1　事件发生图

由此，时间表问题可以描述为求时间表解的问题，即建立事件集E到时空资源集R的映射函数f:

$f : E \to R$

显然，时间表问题的解通常不是唯一的。

1.2.数学模型的建立

排课问题本质上是时间表问题的一类典型应用实例，是为了解决课程安排对时间，空间资源的有效利用并避免发生冲突。所要完成的主要任务是将班级，教室，课程，教师安排在一周内，且不发生时间冲突，此外，还需要考虑课程教学效果，满足教师特殊要求等多项指标。

令时间集合：$T=\{t_1,t_2,t_3,\ldots,t_m\}$；课程集合：$L=\{l_1,l_2,\ldots,l_p\}$；班级集合：$C=\{c_1,c_2,\ldots,c_m\}$；教室集合：$R=\{r_1,r_2,\ldots,r_(m*n)\}$；教师集合：$S=\{s_1,s_2,\ldots,s_k\}$；时间与教室的笛卡尔积$G=T\cdot R=\{(t_1,r_1),(t_1,r_2),\ldots,(t_d,r_n)\}$，G中的元素成为时间-教室对。这样排课问题的求解就转化成为求课程L到时间-教室对G的幂集PS(G)的一个映射：

$$f:L\to PS(G)$$

要排出合理实用的课表，排课过程还必须满足各种约束条件，可以将各种约束条件大体归纳成两类，即软约束和硬约束。

1.2.1硬约束条件：

硬约束条件是指要完成排课问题所必须满足的条件。

（1）同一时间，一个班级不能同时有一门以上的课程，记为Req1:

$$Req1:\sum_{p=1}^{P}\sum_{n=1}^{N}\sum_{k=1}^{K}X(l_pc_mr_ns_kt_d)\le 1$$

其中：$1\le m\le M,1\le d\le D$,值为1表示班级c_m在时间t_d上教师s_k的课程l_p，0表示否。

（2）同一时间，一位教师最多只能同时上一门课程，记为Req2:

$$Req2:\sum_{p=1}^{P}\sum_{n=1}^{N}\sum_{m=1}^{M}X(l_pc_mr_ns_kt_d)\le 1$$

其中：$1\le k\le K,1\le d\le D$,值为1表示教师s_k在时间t_d于教师r_n上课程l_p,0表示否。

（3）同一时间，一个教室不能同时有一门以上的课，记为Req3:

$$Req3:\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{k=1}^{K}X(l_pc_mr_ns_kt_d)\le 1$$

其中：$1\le n\le N,1\le d\le D$,值为1表示教室r_n在时间t_d由教师s_k上课程l_p，0表示否。

1.2.2软约束条件

除了必须满足的硬性约束外，还有软约束条件，这些软约束有助于课表更加合理化，更加人性化。

第五条 课表在编排过程中需遵循以下原则：

（一）按照“先上午、后下午、再晚上”的时序排课，充分用好上午第1-3节课；

（二）按照“必修课先于选修课、合班课先于分班课、公共课先于专业课”顺序排课；

（三）每位教师排课应不超过6课时/天，连续排课一般以不超过3课时/半天为宜；

（四）周课时>3的非实验类课程不得安排在一天内完成，且至少间隔1天；

（五）除部分特殊课程以外，不得从第二节课开始排课；

（六）课程安排应尽量均匀分布在周一至周五，不可过于集中；

（七）周三下午为学校集中安排各类活动时间，原则上不排课。

第六条 教师因客观原因对排课有特殊要求的，须在排课前提出书面申请，由学院（部）酌情考虑。

2.遗传算法设计

遗传算法通过维持一个群体, 并按个体的适应度大小重复地进行选择、交叉和变异等遗传操作来实现群体内个体结构的重组, 将性能良好的解结构遗传下去, 提高后代的适应能力, 从而进化到最优或次优解.其执行的四个步骤:

(1) 随机建立不同可行解组成的初始群体;

(2) 计算各个个体的适应度;

(3) 根据遗传概率, 利用下述操作产生新群体;

复制:利用赌盘法选出优良个体复制添加到新群体中。

交叉:利用赌盘法选出两个个体进行基因交换添加到新群体。

变异:一定概率随机变化其基因添加到新群体。

(4) 判断是否达到终止条件, 达到终止输出最优解, 否则转2。

课表问题的求解过程可以分两个步骤:第一步, 随机可行解的生成.随机可行解是在满足硬约束条件下随机生成的, 即满足约束条件R, 这样保证了生成的解在班级、教师和教室等资源的安排过程不发生冲突;第二步, 对随机可行解进行优化.由于随机可行解的生成是在约束条件R下随机生成的, 并没有考虑排课的软约束条件, 因此所生成的解是非最优解 (劣解) , 需要进一步的优化。

2.1.染色体编码

遗传算法 (GA) 中首要考虑的是如何对染色体编码, 使之适用于操作。 每条染色体用以代表每位教师的课表, 其结构表示如下:

$$(\text{教师}ID, \text{班级}ID, \text{上课时间}, \text{教室编号})$$

在算法设计时, 染色体采用二进制数编码，使用了25位二进制，0-5表示教师，5-10表示班级，10-15表示日期，15-25表示教室，其中教室分为理论和实践，各由五位二进制来表示。

2.2.适应度函数

对每条染色体中存在的冲突类型进行惩罚，如果违背了约束条件，就对该染色体的适应度减去一定的数值，此外对维护了软约束条件的染色体适应度给与一定的奖励，染色体适应度函数值越大, 则表示其拥有越好的授课时段和教室, 在下一代的演化中的生存概率就越大。

$$f = f_{ini} + W * P_i - t * V_i$$

2.3.遗传算法操作

2.3.1初始化

根据实际情况的简化, 假定一个班每周上五天课, 每天有4个授课单元, 则可用行表示20个时间片 (即一个班级一周的课表) , 用列表示班级组成二维数组。对每一班级来说, 首先把有特定教学时间要求的教师-课程编码填入数组, 然后产生一个随机数, 将该班的其他教师-课程编码填入。如果产生的随机数对应的数组变量中已有数据, 则重新产生, 直到将所有教师-课程编码无重复地填入数组。这样就产生了一个初始的课程表, 按种群的大小, 产生一定数量的初始表, 构成初始种群。

2.3.2选择

轮盘赌选择法（roulette wheel selection）是最简单也是最常用的选择方法，在该方法中，各个个体的选择概率和其适应度值成比例，适应度越大，选中概率也越大。设某一部分$x(i)$的适应度值表示为$f(x_i)$，该部分被选中的概率为$p(x_i)$，累积概率为$q(x_i)$，对应的计算公式如下：

$$p(x_i) = \frac{f(x_i)}{\sum_{j=1}^{N} f_i}$$

$$q(x_i) = \sum_{j=1}^{i} p(x_j)$$

2.3.3交叉

交叉时, 根据选择操作的结果选取两条染色体作为父个体, 再取一随机值 (设为r) 与系统预设的交叉率值 (设为t) 比较, 若r<t则进行交换基因。

交叉操作是生成新个体的主要方式，其思想就是将两个个体的部分基因互换产生新基因个体，交叉算子分为很多种，本文采用的是单点交叉算子，随机取染色体上一节点，以该节点为截断点，将染色体分为两段，再将染色体交叉拼接。

2.3.4变异

算法在实现变异时, 分别把染色体的表示教学时间和教学地点的片段进行变异，不改变老师和课程。在实现上, 定义一个变异概率P_m, 在变异时先产生一个随机数r, 当$r < P_m$时, 执行变异操作, 否则不执行。染色体由二进制表示，如果基因的位置是“1”则换为“0”。如果基因的位置是“0”则换为“1”。变异操作是产生新基因的辅助方法，以较小的概率存在于遗传算法中，保证了种群的多样性，防止出现过早收敛。

2.4算法步骤

```
BEGIN:

I = 0;

Initialize P(I);

Fitness P(I);

While(not Terminate-Condition)

{

I ++;

GA-Operation P(I);

Fitness P(I);

}

END.
```

(1) 根据课表时间和资源字典中的教室字典生成“时间←→教室”对, 并以其作为遗传算法中矩阵编码的列.

(2) 生成权值衡量数据库, 记录遗传操作中每个染色体、每位教师编排结果的具体值和每个“时间←→教室”对是否冲突以及冲突的次数.

(3) 根据课表字典的单一资源和开课计划所收集的关系资源, 生成“教师←→课程←→班级”课时对, 并把生成的结果作为矩阵的元素, 教师作为矩阵的行.

(4) 初始化染色体, 并且对其进行适值计算.

(5) 根据值的大小分配一个概率, 通过随机数选择染色体1和染色体2进行行交换操作, 把染色体1中大于染色体2和染色体2中大于染色体1的所有数据作为新生成的染色体3的数据.

(6) 根据值的大小分配一个概率, 通过随机数选择染色体1进行列变异.

(7) 对所有染色体进行适值计算.

(8) 选择最佳的等于群体规模的染色体作为下一代的父代.

(9) 选择出最佳的染色体, 分解染色体, 把染色体的解空间映射为实际问题的解空间, 生成最终的课程表.

3.模型推广及改进

本模型以遗传算法为主要算法思想来进行课程表的编排, 先用“随机匹配”算法来生成遗传算法的初始种群, 解决课表编排的冲突问题, 再利用遗传算法对课程表进行优化, 最后得到一张资源分配最佳组合的排课表。

模型考虑了高校排课的问题，以计算机学院为例完成排课，由于课表问题具有规模大、约束条件复杂、需求不断变化等特征, 本文在排课算法的完善方面还有很多需要努力的地方,缺点较为明显，首先是未严格区分理论课程及实验课程教室，未单独考虑实验教室，并且没有设置硬性约束条件，仅仅以惩罚奖励机制来计算适应度，以此来得到较优解。

想要再提高开课系统的质量还需要在以下几个方面做出改进：

1.单独设置实验教室，区分理论课程实践课程；

2.设置硬性约束条件，有冲突教室课程的解应当剔除；

3.进一步完善惩罚奖励机制，对各软约束条件的权重给与有效赋值。