Автоматизация построения расписания экзаменов ВУЗа с использованием генетического алгоритма

М.Ю. Жукова, В.М. Аль-Габри

Донской государственный технический университет, Ростов-на-Дону

Аннотация: В данной статье рассматривается решение проблемы автоматизированного формирования рационального расписания экзаменов вуза с помощью генетического алгоритма. Предложенный метод учитывает ряд требований необходимых для решения проблемы формирования расписания экзаменов. В данной работе, в процессе формирования начального допустимого расписания экзаменационной сессии, уделяется особое внимание последовательности экзаменов. Для улучшения качества полученного расписания был использован генетический алгоритм, позволяющий минимизировать число нарушений заданных мягких ограничений.
Ключевые слова: генетический алгоритм, расписание, экзамены, ВУЗ, учебная группа.

Введение: В настоящее время достаточно большое количество слушателей различных курсов и образовательных программ по итогам зачисления в учебные заведения, в процессе обучения, а также на этапе квалификации, имеют необходимость пройти через экзаменационные испытания. Процесс укрупнения вузов, реализуемый в настоящее время в РФ, приводит их к значительному территориальному росту, увеличению контингента обучающихся и профессорско-преподавательского состава. Это, в свою очередь, обуславливает рост общего числа экзаменов сессии и увеличивает сложность решения задачи формирования расписаний. Кроме того, задача осложняется наличием у каждого вуза своих специфических требований, что делает разработку универсального метода практически невозможным [1].

Российская образовательная система, традиционно ориентированная на технологию обучения в академических группах, находится только в начале пути индивидуализации. В связи с этим, актуальность решения задачи разработки расписания экзаменов, в процессе развития российского образования, будет только возрастать [2].

Формально расписание экзаменов это отношение, построенное на прямом произведении множеств объектов: учебная группа, преподаватель,

дисциплина [3]. Для решения задачи формирования расписания экзаменов принимается разные методы и технологии, основанные на классических методах и алгоритмах целочисленного программирования, методах раскраски графов. Кроме того, применяются методы: полного перебора, ветвей и границ, а также эвристические методы, в том числе основанные на генетических алгоритмах [4].

Цель данной работы заключается в разработке метода формирования рационального расписания экзаменов в высшем учебном заведении.

Постановка задачи

$$G = \{g_i \,|\, i = \overline{1, n_G}\} = \{(ng, cnt)_i \,|\, i = \overline{1, n_G}\} \quad (1)$$

– множество учебных групп, где *ng* - название группы, *cnt* - количество студентов в группе;

$$T = \{t_i \,|\, i = \overline{1, n_T}\} = \{(id, surn, name, part)_i \,|\, i = \overline{1, n_T}\} \quad (2)$$

– множество преподавателей;

$$Ds = \{ds_i \,|\, i = \overline{1, n_{Ds}}\} \quad (3)$$

– множество дисциплин;

$$DT = \{dt_i \,|\, i = \overline{1, n_{DT}}\} = \{(dt_s, dt_e)_i \,|\, i = \overline{1, n_{DT}}\} \quad (4)$$

– множество диапазонов дат, для проведения экзаменационной сессии.

Множество временных интервалов для расписания экзаменов можно представить в виде:

$$Dtm = \{(\text{date}, \text{tm}) \,|\, date \in dt_i, i = \overline{1, n_{DT}}, \text{tm} = \overline{1, 2}\} \quad (5)$$

где *date* – дата из диапазона $(dt_{start}, dt_{end})_i$, *tm* - временной интервал, принадлежащий дню, когда может быть проведен экзамен (*tm*=1 - первая половина дня, *tm*=2 - вторая половина дня экзамена).

Множество учебных аудиторий можно представить в следующем виде:

$$R = \{r_i \,|\, i = \overline{1, n_R}\} = \{(nr, rns, tr)_i \,|\, i = \overline{1, n_R}\} \quad (6)$$

где *nr* - название аудитории, *rns* - вместимость, *tr* - тип аудитории;

При формировании расписания экзаменов необходимо учитывать ряд требований, предъявляемых к этому расписанию. Требования можно определить как жесткие и мягкие ограничения. Под жесткими ограничениями будем рассматривать требования, нарушение которых приводит к недопустимости использования построенного расписания. К мягким отнесем рекомендации и пожелания [5]. В таблице 1 рассмотрены ограничения с их весовыми коэффициентами.

Таблица 1. Требования к расписанию экзаменов

№	Ограничение	Весовой коэффициент
	1. Жесткие ограничения	
1.1.	В одной учебной группе не может быть более одного экзамена в один и тот же временной интервал	k_{11}
1.2.	Один преподаватель не может проводить более одного экзамена в один и тот же временной интервал	k_{12}
1.3.	Одна аудитория не может быть использована для проведения более чем одного экзамена в один и тот же временной интервал	k_{13}
1.4.	В одной учебной группе минимальный временной промежуток между экзаменами составляет 2 дня	k_{14}
1.5.	Количество посадочных мест в аудитории должно быть больше или равно количеству студентов в группе	k_{15}
	2. Мягкие ограничения	
2.1.	Для учебной группы учитывается порядок следования экзаменов, прогнозирующий наилучшую успеваемость [6]	k_{21}
2.2.	В одной учебной группе минимальный временной промежуток между экзаменами не менее 3 дней	k_{22}
2.3.	Количество свободных посадочных мест в аудитории не превышает 20% от ее общей вместимости	k_{23}
2.4.	Один преподаватель проводит не более одного экзамена в один день	k_{24}

Объемы нарушений ограничений определяются по следующим формулам:

$$c_{11} = \sum_{i=1}^{ng}\sum_{j=1}^{ne} isDEG(\mathrm{g}_i, \mathrm{e}_j) \quad (7)$$

где $isDEG(g_i, e_j) = \begin{cases} 1, & если \quad countExamG(dt_{e_j}, g_i) > 1 \\ 0, & иначе \end{cases}$ - функция, определяющая нарушение ограничения 1.1, *countExamG()* - функция, определяющая количество экзаменов в группе g_i во временном интервале dt_{ej}.

$$c_{12} = \sum_{i=1}^{nt}\sum_{j=1}^{ne} isDET(\mathrm{t}_i, \mathrm{e}_j) \quad (8)$$

где $isDET(\mathrm{t}_i, \mathrm{e}_j) = \begin{cases} 1, & если \quad countExamT(dt_{e_j}, t_i) > 1 \\ 0, & иначе \end{cases}$ - функция, определяющая нарушение ограничения 1.2, *countExamT()* - функция, определяющая количество экзаменов у преподавателя t_i во временном интервале dt_{ej}.

$$c_{13} = \sum_{i=1}^{nr}\sum_{j=1}^{ne} isDER(r_i, e_j) \quad (9)$$

где $isDER(r_i, e_j) = \begin{cases} 1, & если \quad countExamR(dt_{e_j}, r_i) > 1 \\ 0, & иначе \end{cases}$ - функция, определяющая нарушение ограничения 1.3, *countExamR()* - функция, определяющая количество экзаменов в аудитории r_i во временном интервале dt_{ej}.

$$c_{14} = \sum_{i=1}^{ng}\sum_{j=1}^{ne-1} isMinDaysG(\mathrm{g}_i, (\mathrm{e}_j, e_{j+1})) \quad (10)$$

где $isMinDaysG(\mathrm{g}_i, \mathrm{e}_j) = \begin{cases} 1, & если \quad (date_{e_{j+1}} - date_{e_j} - 1) < 2 \\ 0, & иначе \end{cases}$ - функция, определяющая нарушение ограничения 1.4.

$$c_{15} = \sum_{i=1}^{ng}\sum_{j=1}^{ne} isRoomCap(g_i, e_j) \quad (11)$$

где $isRoomCap(g_i, e_j) = \begin{cases} 1, & если\ countst(g_i) > RoomCap(r_{e_j}) \\ 0, & иначе \end{cases}$ - функция, определяющая нарушение ограничения 1.5, *countst()* - функция, определяющая количество студентов в группе, *RoomCap()* - функция, определяющая количество посадочных мест в аудитории.

$$c_{21} = \sum_{i=1}^{ng} isConsExm(g_i) \quad (12)$$

где $isConsExm(g_i) = 1 - getWConsExm(g_i)$ - функция, определяющая нарушение ограничения 2.1, $getWConsExm(g_i) \in [0;1]$ - функция получения коэффициента успеваемости группы по заданной последовательности экзаменов (w_i).

$$c_{22} = \sum_{i=1}^{ng} \sum_{j=1}^{ne} is\,\mathrm{Re}comDaysG(g_i, (e_j, e_{j+1})) \quad (13)$$

где $is\,\mathrm{Re}comDaysG(g_i, (e_j, e_{j+1})) = \begin{cases} 1, & если\ (date_{e_{j+1}} - date_{e_j} - 1) < 3 \\ 0, & иначе \end{cases}$ - функция, определяющая нарушение ограничения 2.2.

$$c_{23} = \sum_{i=1}^{ng} \sum_{j=1}^{ne} is\,\mathrm{Re}comRoomCap(g_i, e_j) \quad (14)$$

где $is\,\mathrm{Re}comRoomCap(g_i, e_j) = \begin{cases} \dfrac{roomcap(r_{e_j}) - countst(g_i)}{roomcap(r_{e_j})} - i\,\mathrm{Re}com, & если\ \dfrac{roomcap(r_{e_j}) - countst(g_i)}{roomcap(r_{e_j})} > i\,\mathrm{Re}com \\ 0, & если\ \dfrac{roomcap(r_{e_j}) - countst(g_i)}{roomcap(r_{e_j})} \le i\,\mathrm{Re}com \end{cases}$ - функция, определяющая нарушение ограничения 2.3, *countst()* - функция, определяющая количество студентов в группе, *RoomCap()* - функция, определяющая количество посадочных мест в аудитории, *iRecom* - рекомендуемый максимальный процент свободных мест в аудитории.

$$c_{24} = \sum_{i=1}^{nt} \sum_{j=1}^{ne} is\,\mathrm{Re}comET(t_i, e_j) \quad (15)$$

где $is\,\mathrm{Re}comET(t_i, e_j) = \begin{cases} 1, & если\ countExamT(date_{e_j}, t_i) > 1 \\ 0, & иначе \end{cases}$ - функция, определяющая нарушение ограничения 2.4, *countExamT()* - функция, определяющая количество экзаменов у преподавателя t_i в день экзамена e_j ($date_{ej}$).

Для определения качества сформированного расписания экзаменов используется целевая функция, представляющая собой количественную оценку нарушений ограничений [7].

$$F(x)=\sum_{i=1}^{n}\sum_{j=1}^{m_i}k_{ij}\times c_{ij}(x) \quad (16)$$

где k_{ij} – весовой коэффициент ограничения, c_{ij} – объем нарушений ij-го ограничения, n – число типов ограничений (в данном случае n =2), m_i - количество ограничений i-го типа, x - текущее решение.

Алгоритм построения допустимого расписания

При построении допустимого расписания основополагающим фактором является использование заранее известной последовательности экзаменов, прогнозирующей наилучшую успеваемость группы.

$aProgress = \{(g_i, (e_1, e_2, \ldots, e_{ne}, w)_j) \mid g_i \in G, j=\overline{1,n_i}!, w\in[0, 1]\}$ – массив зависимостей успеваемости группы от порядка следования экзаменов, отсортированные по убыванию w, e_l – экзамен с порядковым номером l, w - коэффициент успеваемости группы.

$(e_1, e_2, \ldots, e_{ne})_{jProgress} = getSequence(g_i, jProgress)$ – функция, получающая для группы g_i, по порядковому номеру $jProgress$ последовательность экзаменов из массива $aProgress$.

$minTimeout$ - минимально количество дней между экзаменами.

$normTimeout$ - рекомендуемое количество дней между экзаменами.

$countApptExams$ - количество расставленных экзаменов в группе.

$Q_{allnappt}$ - очередь из всех нераспределенных экзаменов.

$eCount()$ - функция вычисления количества экзаменов в группе.

$maxTOg_i$ - максимально допустимый интервал между экзаменами в период заданного диапазона.

$getCountApptExams()$ - функция получения количества расставленных экзаменов.

$addExamToQ()$ - функция добавления экзамена в очередь.

$removeExamFromQ()$ - функция удаления экзамена из очереди.

isApptExam() - функция проверки расставлен ли экзамен (*true* - расставлен, *false* - не расставлен).

setRDExam() - функция присвоения пространственно-временной характеристика экзамену.

getPossibleDate() - функция получения возможной даты из диапазона дат для проведения экзамена.

getNextDate() - функция получения следующей даты.

Псевдокод алгоритма построения допустимого расписания экзаменов:

1 **СОРТИРОВКА** групп по диапазонам сессии.
2 **ЦИКЛ** по группам $g_i \in G$
3 **ВЫЧИСЛИТЬ** $eCount(g_i)$
4 **ВЫЧИСЛИТЬ** $maxTOg_i$ с округлением в меньшую сторону
5 **ЕСЛИ** ($maxTOg_i < minTimeout$)
6 **ТО** переход на шаг 2
7 **ИНАЧЕ**
8 **ЕСЛИ** $getCountApptExmas(g_i) == eCount(g_i)$
9 **ТО** переход на шаг 2
10 **ИНАЧЕ**
11 **ПРИСВАИВАЕМ** $jProgress = 1$
12 **ПРИСВАИВАЕМ** $Q_{jpgi} = getSequence(g_i, jProgress)$
13 **ПРИСВАИВАЕМ** $countApptExmas[g_i] = 0$
14 **ЦИКЛ ПОКА** не конец очереди Q_{jpgi}
15 **ЕСЛИ** $isApptExam(e_j) == true$
16 **ТО**
17 $removeExamFromQ(e_j, Q_{jpgi})$
18 $countApptExmas[g_i]+1$
19 переход на шаг 14
20 **ИНАЧЕ**
21 $date = getPossibleDate(g_i)$
22 **ЕСЛИ** $date == null$
23 **ТО** $addExamToQ(e_j, Q_{allnappt})$

ИНАЧЕ
ЕСЛИ *findConf*(*date*, *g*, e_j) == *false*
ТО
setRDExam(*date*, *g*, e_j)
countApptExmas[g_i] +1
removeExamFromQ(e_j,Q_{jpgi})
ИНАЧЕ
date = *getNextDate*()
ЕСЛИ *date* == *null*
ТО
jProgress+1
переходим на шаг 12
ИНАЧЕ переход на шаг 25
ВСЕ ЕСЛИ
ВСЕ ЕСЛИ
ВСЕ ЕСЛИ
ВСЕ ЕСЛИ
ВСЕ ЦИКЛ ПОКА
ВСЕ ЕСЛИ
ВСЕ ЕСЛИ
ВСЕ ЦИКЛ

Минимизация целевой функции (16) реализуется с помощью генетического алгоритма, основанного на механизмах естественного отбора и наследования[8].

Будем рассматривать расписание экзаменационной сессии как некоторое однозначное отображение *ES* из множества экзаменов *E* во множество *Dtm* × *R* (декартово произведение временных интервалов и аудиторий):

$$ES : E \rightarrow Dtm \times R.$$

Табличное представление данного отображения представлено в таблице 2.

Таблица 2. Табличное представление отображения *ES*

date	***tm***	***r1***	***r2***	***...***	***...***	***rnr***
d1	*1*	*e1*	*e2*	...	...	*el*
d1	*2*	X	X	...	...	...
d2	*1*	...	*em*	...	...	...
d2	*2*	...	...	...	*ek*	X
...	...	...	...	...	...	...
dt	*1*	...	...	...	...	...
dt	*2*	*ei*	X	...	*ej*	...

Так как размерность *ndr* множества *Dtm* × *R* (будем называть это множество множеством пространственно-временных характеристик) обычно превышает количество экзаменов *ne*, то некоторые пространственно-временные характеристики будут являться свободными (в таблице такие позиции обозначены X).

Если пронумеровать все ячейки таблицы 1, и развернуть их в строку, то получится последовательность пространственно-временных характеристик с заданными на них экзаменами. Представление допустимого расписания в таком виде, позволяет использовать ГА для улучшения этого решения. В нашем случае, в терминологии ГА, под геном будет пониматься ячейка таблицы расписания, а под особью или хромосомой - один из вариантов расписания экзаменов. При этом «жизнеспособными» будут считаться только те хромосомы, которые удовлетворяют заданным ограничениям, т.е. являются допустимыми решениями.

Популяцией будем считать множество «жизнеспособных» хромосом с наиболее высокой степенью выживания (с наибольшим значением функции «приспособленности»).

Целесообразно упорядочить множество хромосом по убыванию функции «приспособленности», чтоб создание новой популяции начинать с «лучших» хромосом. Однако такая комбинация не всегда дает лучший

результат, поэтому при генерации новых популяций необходимо рассмотреть и комбинации скрещивания с «менее приспособленными» особями.

Множество допустимых решений поставленной задачи будет являться начальной популяцией для работы генетического алгоритма.

Скрещивание (или кроссовер)

Наиболее продуктивным методом скрещивания (кроссовера), является метод основанный на циклических перестановках.

Суть метода заключается в следующем:

1. Выбираются 2 родительские хромосомы, которые будут участвовать в скрещивании (CH1 и CH2). Например:

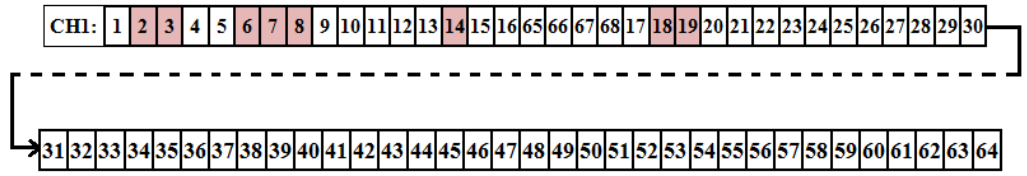

Рисунок 1. - Родительская хромосома CH1.

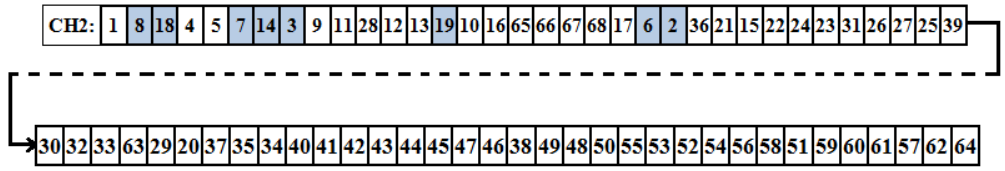

Рисунок 2. - Родительская хромосома CH2.

2. Построим циклические перестановки по родительским хромосомам:

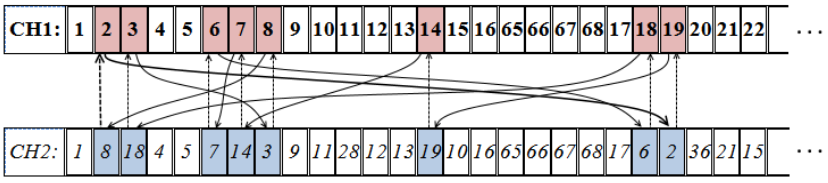

Рисунок 3. - Пример построения циклических перестановок.

При разбиении хромосомы на циклические перестановки могут возникнуть циклы единичной длины (тривиальные циклы). Обмен такими циклами не несет никакой функциональной нагрузки, поэтому для удобства и

простоты описания хромосомы будем обозначать через Z_i последовательность из таких циклов.

CH1 = Z_1+$CH1_1$+Z_2+$CH1_2$+Z_3+$CH1_3$+Z_4+$CH1_4$+Z_5+$CH1_5$+Z_6

CH2 = Z_1+$CH2_1$+Z_2+$CH2_2$+Z_3+$CH2_3$+Z_4+$CH2_4$+Z_5+$CH2_5$+Z_6

где Z_1 = (1), Z_2 = (4)(5)(9), Z_3 = (12)(13)(16)(65)(66)(67)(68)(17),

Z_4 = (21)(24)(32)(33)(37)(40)(41)(42)(43)(44)(45), Z_5 = (49),

Z_6 = (53)(54)(56)(59)(60)(61)(64)

$CH1_1$ = (2,19,14,7,6,18,3,8), $CH2_1$ = (8,3,18,6,7,14,19,2),

$CH1_2$ = (10,15,22,23,25,29,35,38,48,50,51,58,57,62,63,34,39,30,31,26,27,28,11),

$CH2_2$ = (11,28,27,26,31,30,39,34,63,62,57,58,51,50,48,38,35,29,25,23,22,15,10),

$CH1_3$ = (20,36), $CH2_3$ = (36,20),

$CH1_4$ = (46,47), $CH2_4$ = (47,46),

$CH1_5$ = (52,55), $CH2_5$ = (55,52).

3. Операция скрещивания может быть представлена как обмен совпадающих по набору генов циклов:

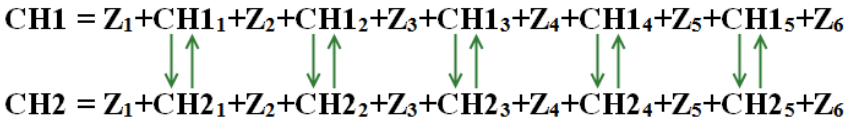

Рисунок 4. - Пример выполнения операции скрещивания.

Тогда хромосомы-потомки будут представлять собой комбинацию вида:

$$\mathbf{Z_1}+ X_1+\mathbf{Z_2}+ X_2+\mathbf{Z_3}+ X_3+\mathbf{Z_4}+ X_4+\mathbf{Z_5}+ X_5+\mathbf{Z_6},$$

где X_i - i-й цикл родительской хромосомы CH1 или CH2.

Количество хромосомы-потомки будет равно 2^{nc}-2 (где nc – количество нетривиальных циклов в родительских хромосомах).

Таблица 3. Возможные хромосомы-потомки

CH1:	Z_1+$CH1_1$+Z_2+$CH1_2$+Z_3+$CH1_3$+Z_4+$CH1_4$+Z_5+$CH1_5$+Z_6
CH2:	Z_1+**$CH2_1$**+Z_2+**$CH2_2$**+Z_3+**$CH2_3$**+Z_4+**$CH2_4$**+Z_5+**$CH2_5$**+Z_6
Потомки:	Z_1+**$CH2_1$**+Z_2+$CH1_2$+Z_3+$CH1_3$+Z_4+$CH1_4$+Z_5+$CH1_5$+Z_6
	Z_1+$CH1_1$+Z_2+**$CH2_2$**+Z_3+$CH1_3$+Z_4+$CH1_4$+Z_5+$CH1_5$+Z_6
	…
	Z_1+$CH1_1$+Z_2+**$CH2_2$**+Z_3+**$CH2_3$**+Z_4+**$CH2_4$**+Z_5+$CH1_5$+Z_6
	Z_1+$CH1_1$+Z_2+**$CH2_2$**+Z_3+**$CH2_3$**+Z_4+$CH1_4$+Z_5+**$CH2_5$**+Z_6

	…
	Z_1+**CH2$_1$**+Z_2+**CH2$_2$**+Z_3+**CH2$_3$**+Z_4+CH1$_4$+Z_5+**CH2$_5$**+Z_6
	Z_1+**CH2$_1$**+Z_2+**CH2$_2$**+Z_3+**CH2$_3$**+Z_4+**CH2$_4$**+Z_5+CH1$_5$+Z_6

Возможность получения большого количества хромосом-потомков увеличивает вероятность нахождения оптимального решения (хромосом с наилучшей функцией приспособленности).

Мутация

Генетический оператор мутации изменяет значение гена на противоположное (в бинарном представлении хромосом 0 на 1 или наоборот) с некоторой вероятностью p_m.

Например, в хромосоме CH1 5-й ген представляет кортеж $\{d, e_i, r_j\}$, а 14-й ген: $\{d, e_k, r_l\}$. При этом аудитории r_j и r_l являются идентичными. Под идентичными, будем понимать аудитории, имеющие аналогичное материально-техническое оснащение и вместимость. Тогда мутация в хромосоме CH1 может быть представлена на рис.4.

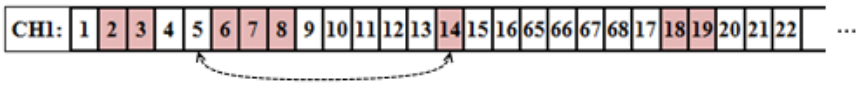

Рисунок 4. - Пример мутации в хромосоме CH1.

Функция приспособленности

Чтобы оптимизировать какую-либо структуру с использованием ГА, нужно задать меру качества для каждого индивида в пространстве поиска. Для этой цели используется функция приспособленности. В поставленной задаче целевая функция (16) выступает в качестве функции приспособленности [9,10].

Результаты

Разработанный экспериментальный программный стенд позволяет:

- проанализировать существующие расписания;

- исключить пространственно-временные накладки в разнотипных расписаниях, сформированных в разных системах, но загруженных в общую базу;
- построить, по допустимым расписаниям, квазиоптимальное по использованию аудиторного фонда.

Выводы

1. Актуальность поставленной задачи обусловлена появлением в университетах большого количества новых образовательных программ в условиях ограниченности аудиторного фонда.
2. В связи с переходом российской образовательной системы к расширению использования индивидуальных траекторий обучения, потребность решения задач автоматизации для построения расписания экзаменов будет возрастать.
3. Задачи построения расписаний занятий и экзаменов сходны по постановке, целям и методам решения и отличаются по размерности. размерность задачи построения расписания занятий, в общем случае, выше, чем значение этого параметра в задаче построения расписания экзаменов.
4. Использование механизмов построения расписаний экзаменов совместно с системами построения рациональных последовательностей экзаменов, позволит проектировать системы автоматизации построения расписаний сессий и управления их результатами.

Литература

1. Дворянкин А.М. Обзор методов составления расписания вузов/ А.М. Дворянкин, В.С. Чалышев // Изв. ВолгГТУ. Серия Актуальные проблемы управления, вычислительной техники и информатики в технических

системах: межвуз. сб. науч. ст. / ВолгГТУ.- Волгоград, 2011. - Вып. 11, № 9. - с. 110-113.

2. Аль-Габри В.М., Обзор литературных источников по теме «Автоматизация составления расписания занятий и экзаменов в высших учебных заведениях»// Вестник Донского государственного технического университета 2017, №1(88), с. 132-143.

3. Гранков М.В., Аль-Габри В.М., Горлова М.Ю., Анализ и кластеризация основных факторов, влияющих на успеваемость учебных групп вуза// Инженерный вестник Дона, 2016, № 4, URL: ivdon.ru/ru/magazine/archive/n4y2016/3775.

4. Семенов С.П. Сравнительный анализ подходов к автоматизации составления расписаний учебных занятий в образовательных учреждениях / С.П. Семенов, Я.Б. Татаринцев // Изв. Алтайского гос. ун-та. – 2010. - № 1(65). – с. 103-105.

5. Абухания А.Ю. Модели, алгоритмы и программные средства обработки информации и принятия решений при составлении расписаний занятий на основе эволюционных методов: Дис. …канд. техн. наук: 05.13.01 / Абухания Амер Юсеф – Новочеркасск, 2016. – 231 с.

6. Гранков М.В., Аль-Габри В.М., Регрессионная модель успеваемости студенческих групп вуза// Инженерный вестник Дона, 2017, № 1, URL: ivdon.ru/ru/magazine/archive/ /n1y2017/4058

7. Omar Ibrahim Obaid, MohdSharifuddin Ahmad, Salama A. Mostafa, Mazin Abed Mohammed. Comparing Performance of Genetic Algorithm with Varying Crossover in Solving Examination Timetabling Problem/ Omar I.O, MohdSharifuddin A., Salama A.M., Mazin A.M. // Journal of Emerging Trends in Computing and Information Sciences- VOL. 3, NO.10, Oct 201, pp. 1427- 1434.

8. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский. - М.: Горячая линия – Телеком, 2013. - 384 с.

9. Burke E. A Genetic Algorithm for university timetabling // In AISB Workshop on Evolutionary Computing. University of Leeds. - UK. - 1994. – pp. 134 – 140.

10. Лопатеева О.Н. Система автоматизированного формирования учебного расписания в высшем учебном заведении на основе эвристических алгоритмов: Дис. …канд. техн. наук: 05.13.01 / Лопатеева Ольга Николаевна – Красноярск, 2006. – 205 с.

References

1. Dvoryankin A.M. Izv. VolgGTU. Seriya Aktual'nyye problemy upravleniya, vychislitel'noy tekhniki i informatiki v tekhnicheskikh sistemakh: mezhvuz. sb. nauch. st. VolgGTU.Volgograd, 2011. Vyp. 11, № 9, pp. 110-113.

2. Al'-Gabri V.M., Vestnik Donskogo gosudarstvennogo tekhnicheskogo universiteta 2017, №1 (88), pp. 132-143.

3. Grankov M.V., Al'-Gabri V.M., Gorlova M.U., Inženernyj vestnik Dona (Rus), 2016, № 4, URL: ivdon.ru/ru/magazine/archive/n4y2016/3775.

4. Semenov S.P. Izv. Altayskogo gos. un-ta. 2010. № 1(65). pp. 103-105.

5. Abukhaniya A.YU. Modeli, algoritmy i programmnyye sredstva obrabotki informatsii i prinyatiya resheniy pri sostavlenii raspisaniy zanyatiy na osnove evolyutsionnykh metodov [Models, algorithms and software for information processing and decision making in scheduling lessons based on evolutionary methods] : Dis. kand. tekhn. nauk: 05.13.01. Abukhaniya Amer Yusef. Novocherkassk, 2016. 231 p.

6. Grankov M.V., Al'-Gabri V.M., Inženernyj vestnik Dona (Rus), 2017, № 1, URL: ivdon.ru/ru/magazine/archive/n4y2017/3775.

7. Omar Ibrahim Obaid, MohdSharifuddin Ahmad, Salama A. Mostafa, Mazin Abed Mohammed. Comparing Performance of Genetic Algorithm with Varying Crossover in Solving Examination Timetabling Problem. Omar I.O, MohdSharifuddin A., Salama A.M., Mazin A.M. Journal of Emerging Trends in Computing and Information Sciences. VOL. 3, NO.10, Oct 2012, pp. 1427- 1434.

8. Rutkovskaya D. Neyronnyye seti, geneticheskiye algoritmy i nechetkiye sistemy [Neural networks, genetic algorithms and fuzzy systems]. M.: «goryachaya linya-telekom», 2013. 384 p.

9. Burke E. A Genetic Algorithm for university timetabling. In AISB Workshop on Evolutionary Computing. University of Leeds. UK. 1994, pp. 134 – 140.

10. Lopateyeva O.N. Sistema avtomatizirovannogo formirovaniya uchebnogo raspisaniya v vysshem uchebnom zavedenii na osnove evristicheskikh algoritmov [The system of the automated formation of the educational schedule in a higher educational institution on the basis of heuristic algorithms]: Dis. kand. tekhn. nauk: 05.13.01. Lopateyeva Ol'ga Nikolayevna. Krasnoyarsk, 2006. 205 p.