ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Апрель–июнь 2023. Т. 30, № 2. С. 48–66

УДК 519.85 DOI 10.33048/daio.2023.30.735

О ЗАДАЧЕ СОСТАВЛЕНИЯ РАСПИСАНИЯ РАБОТЫ ОПЕРАТОРОВ ЦЕНТРА ОБРАБОТКИ ВЫЗОВОВ

А. В. Еремеев[a], *М. А. Сахно*[b]

Институт математики им. С. Л. Соболева, Омский филиал,
ул. Певцова, 13, 644099 Омск, Россия
E-mail: [a] eremeev@ofim.oscsbras.ru, [b] maxim.sakhno@gmail.com

Аннотация. Работа посвящена решению задачи составления расписаний работы персонала центра обработки вызовов. Сформулирована модель целочисленного линейного программирования, доказана труднорешаемость задачи, предложен генетический алгоритм, учитывающий специфику задачи, а также проведено экспериментальное сравнение оптимальных решений, полученных с помощью пакета CPLEX, с решениями, найденными генетическим алгоритмом. Вычислительный эксперимент показал практически приемлемую точность решений, полученных генетическим алгоритмом, и его применимость к задачам большой размерности. Табл. 1, библиогр. 18.

Ключевые слова: центр обработки вызовов, целочисленное линейное программирование, генетический алгоритм, вычислительная сложность.

Введение

Эффективное использование трудовых ресурсов имеет высокую значимость для компаний, желающих снизить расходы на оплату труда и в то же время повысить качество работы своих сотрудников. В связи с этим разработка и использование методов управления персоналом приобретают особую актуальность. Настоящая работа посвящена построению расписаний работы операторов центра обработки вызовов, где операторы осуществляют обработку входящих вызовов, например, для центра техподдержки или горячей линии.

Построению расписаний работы операторов центра обработки вызовов, специализирующегося на обработке входящих вызовов только одного типа, уделяется много внимания в литературе. Как правило, рабочая

Исследование выполнено за счёт гранта Российского научного фонда (проект № 21–41–09017).

нагрузка значительно варьируется в течение дня [1–3], поэтому принято делить день на несколько периодов, в течение которых назначение операторов остаётся постоянным, а изменениями частоты поступления вызовов можно пренебречь. В современных центрах обработки вызовов длительность периодов обычно составляет 15 минут и менее.

Задачи построения расписаний работы операторов центра обработки вызовов часто решаются в два этапа [4]: на первом этапе для каждого периода определяется необходимое число операторов для обработки входящих вызовов, а на втором — рассчитывается набор смен, который с минимальными затратами покрывает эту потребность в персонале.

Задача построения расписания является более сложной в случае, когда каждый оператор может обрабатывать вызовы нескольких типов, поскольку возникает большее число вариантов назначений. В работе [5] предложена методика оптимального укомплектования центра обработки вызовов операторами для одного интервала времени в случае нескольких типов вызовов. В [6] предложено решение той же проблемы с использованием методов локального поиска в сочетании с аналитической аппроксимацией уровня сервиса и локальным улучшением с помощью имитационного моделирования на финальном этапе. В [7] накладывается ограничение только на совокупный уровень сервиса (для всех типов вызовов). В [8] рассматривается вариант обобщённой задачи о назначениях, где также имеется несколько типов работ, но отсутствует фактор времени.

Для задачи построения расписания в случае многих типов вызовов в [9] предложен двухэтапный подход, в котором на первом этапе определяется необходимое число операторов для каждого периода, а на втором — рассчитывается расписание путём решения задачи целочисленного программирования. В [10] для решения этой задачи предложен алгоритм, основанный на имитационном моделировании. Этот алгоритм расширяет метод, предложенный в [5], который решает задачу укомплектования центра обработки вызовов операторами на один период.

В задачах, рассмотренных выше, для каждого оператора известно, способен ли он обрабатывать вызовы некоторого типа, в то время как в современных центрах обработки вызовов часто используется система навыков, в которой отражено, насколько качественно оператор справляется с обработкой вызовов некоторого типа. Как правило, значение навыка выражается целым числом, где большее значение означает более высокое качество обработки вызовов некоторого типа. Система навыков учитывается в задаче, рассматриваемой в данной статье. Также рассматриваемая задача, в отличие от перечисленных выше, имеет ограничение на фонд оплаты труда, так как многие организации имеют ограниченный бюджет на оплату работы операторов. Заметим также, что стоимость

работы оператора зависит от многих факторов, таких как, например, его навыки, время суток (стоимость работы в ночное время может быть выше, чем в дневное) и т. д. Данный параметр также учитывается в настоящей работе.

Для рассматриваемой задачи в данной работе сформулирована модель целочисленного линейного программирования (ЦЛП), доказана её NP-трудность, предложен генетический алгоритм и представлены результаты вычислительных экспериментов.

Работа состоит из трёх разделов: в первом описаны постановка задачи и её модель ЦЛП, во втором содержится описание генетического алгоритма, а в третьем представлены процесс генерации данных и результаты тестирования алгоритма.

1. Постановка задачи и её математическая модель

1.1. Постановка задачи. Имеется центр обработки вызовов, в котором операторы обрабатывают входящие вызовы разных типов. Для каждого оператора известно, какие типы вызовов и насколько качественно он может обрабатывать. Качество обработки вызова оператором определяется его навыком, выраженным целым числом (как правило, навык определяется на основе экспертных оценок и статистических данных). Время рабочего дня разделено на интервалы фиксированной длины. Для каждого оператора известно, в какие интервалы времени он может работать и в какие ему может быть назначен перерыв на обед, причём продолжительность перерыва для всех операторов одинаковая. В течение рабочего дня каждому оператору должен быть назначен ровно один перерыв, другие перерывы в работе не допускаются.

В те интервалы времени, в которые оператор находится на работе, он может быть назначен на обработку вызовов определённого типа. В разные интервалы времени тип обрабатываемых вызовов для одного оператора может меняться. Также для каждого оператора заданы минимальное и максимальное число интервалов времени, в которые он должен быть назначен на обработку вызовов. В каждый интервал времени для каждого типа вызова известно необходимое число операторов для обработки вызовов этого типа, представляющее собой прогнозное значение. Также заданы стоимости работы индивидуально для каждого оператора в различные интервалы времени, причём количество средств, которое можно потратить на оплату труда задействованных операторов, ограничено.

Под расписанием будем понимать назначение каждому оператору его рабочего времени и перерыва, а также типов обрабатываемых вызовов в различные интервалы времени. Необходимо составить расписание работы операторов центра обработки вызовов, максимизирующее *качество*

обслуживания, где под качеством обслуживания понимается среднее значение навыков задействованных операторов в обработке тех типов вызовов, на которые они назначены. При этом в каждый интервал времени для каждого типа вызова должно быть назначено требуемое число операторов.

У операторов в рабочее время имеется только один перерыв, а требуемое число операторов для каждого типа вызова меняется в течение дня, поэтому может возникнуть ситуация, в которой в некоторые интервалы времени какая-то часть операторов может находиться на работе, но не быть назначенными на обработку вызовов. Такие операторы составляют резерв и могут быть назначены в реальном времени, если в действительности для какого-либо типа вызова потребуется больше операторов, чем прогнозировалось.

Введём следующие обозначения:

n — число операторов,

m — число типов вызовов,

k — число интервалов времени,

$P = \{1, \ldots, n\}$ — множество операторов,

$C = \{1, \ldots, m\}$ — множество типов вызовов,

$T = \{1, \ldots, k\}$ — множество интервалов времени,

H_p — минимальное число интервалов времени, в которые оператор $p \in P$ назначен на обработку вызовов,

G_p — максимальное число интервалов времени, в которые оператор $p \in P$ назначен на обработку вызовов,

D — число интервалов времени на перерыв,

S — максимальная оценка качества обработки вызова оператором,

F — объём фонда оплаты труда операторов,

$m_{pc} = 1$, если оператор $p \in P$ может обработать вызов типа $c \in C$, иначе $m_{pc} = 0$,

$w_{pt} = 1$, если оператор $p \in P$ может работать в интервал времени $t \in T$, иначе $w_{pt} = 0$,

$d_{pt} = 1$, если оператору $p \in P$ может быть назначен перерыв в интервал времени $t \in T$, иначе $d_{pt} = 0$,

$s_{pc} \in \{0, 1, \ldots, S\}$ — навык оператора $p \in P$ в обработке вызова типа $c \in C$,

c_{pt} — стоимость работы оператора $p \in P$ в интервал времени $t \in T$,

n_{ct} — число операторов, достаточное для обслуживания вызовов типа $c \in C$ в интервал времени $t \in T$.

При этом интервалы, когда оператор может работать и когда ему может быть назначен перерыв, задаются таким образом, что перерыв не может быть назначен слишком близко к началу или окончанию работы.

1.2. Математическая модель. Для построения модели введём следующие булевы переменные:

$x_{pct} = 1$, если и только если оператор $p \in P$ назначен на обработку вызова типа $c \in C$ в интервал времени $t \in T$,

$y_{pt} = 1$, если и только если оператор $p \in P$ работает в интервал времени $t \in T$,

$z_{pt} = 1$, если и только если оператор $p \in P$ начал работать в интервал времени $t \in T$,

$q_{pt} = 1$, если и только если оператору $p \in P$ назначен перерыв в интервал времени $t \in T$,

$r_{pt} = 1$, если и только если оператор $p \in P$ в интервал времени $t \in T$ выходит на перерыв.

Модель целочисленного линейного программирования для рассматриваемой задачи может быть записана следующим образом:

$$\sum_{p\in P}\sum_{c\in C}\sum_{t\in T} s_{pc}x_{pct} \to \max \tag{1}$$

$$y_{pt} \leqslant w_{pt}, \quad p \in P,\ t \in T, \tag{2}$$

$$y_{pt} \leqslant 1 - q_{pt}, \quad p \in P,\ t \in T, \tag{3}$$

$$\sum_{t\in T} z_{pt} \leqslant 1, \quad p \in P, \tag{4}$$

$$(y_{pt} + q_{pt}) - (z_{p(t-1)} + l_{p(t-1)}) \leqslant z_{pt}, \quad p \in P,\ t \in T \setminus \{1\}, \tag{5}$$

$$q_{pt} \leqslant d_{pt}, \quad p \in P,\ t \in T, \tag{6}$$

$$\sum_{t\in T} r_{pt} = \sum_{t\in T} z_{pt}, \quad p \in P \tag{7}$$

$$q_{pt} - l_{p(t-1)} \leqslant r_{pt}, \quad p \in P,\ t \in T \setminus \{1\}, \tag{8}$$

$$\sum_{t\in T} q_{pt} = D, \quad p \in P, \tag{9}$$

$$x_{pct} \leqslant m_{pc}, \quad p \in P,\ c \in C,\ t \in T, \tag{10}$$

$$\sum_{c\in C}\sum_{t\in T} x_{pct} \geqslant H_p, \quad p \in P, \tag{11}$$

$$\sum_{c\in C}\sum_{t\in T} x_{pct} \leqslant G_p, \quad p \in P, \tag{12}$$

$$\sum_{p\in P} x_{pct} = n_{ct}, \quad c \in C,\ t \in T, \tag{13}$$

$$\sum_{p\in P}\sum_{t\in T} c_{pt}y_{pt} \leqslant F, \tag{14}$$

$$x_{pct} \leqslant y_{pt}, \quad p \in P,\ c \in C,\ t \in T, \tag{15}$$

$$x_{pct}, y_{pt}, z_{pt}, q_{pt}, r_{pt} \in \{0, 1\}, \quad p \in P,\ c \in C,\ t \in T. \tag{16}$$

Целевая функция (1) задаёт критерий максимизации качества обслуживания. Неравенство (2) позволяет назначить оператора на работу только тогда, когда он может работать. Ограничение (3) не позволяет работать оператору во время перерыва, а (4) не позволяет оператору выйти на работу более одного раза. Ограничение (5) позволяет оператору иметь ровно один перерыв. Условие (6) позволяет назначить оператору перерыв только в допустимые интервалы времени, а (7) гарантирует, что оператору будет назначен перерыв только тогда, когда он вышел на работу. Ограничение (8) задаёт непрерывность перерыва, (9) гарантирует, что длительность перерыва будет составлять D интервалов времени, (10) позволяет назначать оператора на обработку только тех типов вызовов, которые он способен обработать. Условия (11) и (12) гарантируют соблюдение минимального и максимального числа интервалов времени, когда оператор назначен на обработку вызовов. Ограничение (13) гарантирует достаточное число операторов для обработки входящих вызовов, (14) не позволяет потратить большее количество средств на оплату труда операторов чем F, а (15) позволяет назначить оператора на обработку вызовов только тогда, когда он работает.

1.3. Труднорешаемость задачи.

Теорема 1. *Задача составления расписания работы операторов центра обработки вызовов* NP*-трудна.*

Доказательство. Сформулируем указанную задачу в форме верификации свойств. В качестве входных данных будем рассматривать те же данные, что и в исходной задаче, но дополненные рациональным числом Q. Задача верификации свойств состоит в ответе на вопрос: *существует ли расписание работы операторов центра обработки вызовов со значением целевой функции не менее Q?*

Ясно, что сформулированная в форме верификации свойств задача принадлежит классу NP. Сведём к ней следующую классическую NP-полную задачу РАЗБИЕНИЕ (см., например, [11]).

Задача РАЗБИЕНИЕ. *Дано n' натуральных чисел α_j, $j = 1, \ldots, n'$. Вопрос: существует ли подмножество $\mathcal{I} \subset \{1, \ldots, n'\}$ такое, что*

$$\sum_{i \in \mathcal{I}} \alpha_i = \frac{1}{2} \sum_{i=1}^{n'} \alpha_i ?$$

Пусть число интервалов времени k равно n', число операторов n равно $2n'$, число типов входящих вызовов m равно n'. В интервал времени $i \in \{1, \ldots, n'\}$ для типа входящего вызова i требуется ровно один оператор для его обработки, а в остальные интервалы времени — нуль.

Тип входящего вызова i могут обработать только операторы $2i-1$ и $2i$, причём у оператора $2i$ величина навыка для вызова типа i, а также стоимость в момент времени i равны 0. Кроме того, положим число интервалов времени на перерыв D равным 0, а минимальное H_p и максимальное G_p число интервалов времени работы оператора $p \in P$ равными 0 и n' соответственно. Дополнительно положим $w_{pt} = 1$ и $d_{pt} = 1$ для всех $p \in P$ и $t \in T$.

Парам операторов $2i-1$ и $2i$ сопоставим один элемент α_i входных данных задачи РАЗБИЕНИЕ так, что выбор оператора $2i-1$ будет соответствовать включению числа α_i в подмножество $\mathcal{I}$, а выбор оператора $2i$ — пропуску числа α_i (по построению в каждый интервал i работает ровно один из двух операторов $2i-1$ и $2i$). Величину навыка оператора $2i-1$ и стоимость работы в интервал времени i положим равными α_i, а верхнюю границу F на фонд оплаты труда — равной $\frac{1}{2}\sum\limits_{i=1}^{n'}\alpha_i$.

Если в задаче РАЗБИЕНИЕ существует искомое множество $\mathcal{I}$, то легко убедиться, что существует допустимое решение, где средний навык при обслуживании вызовов составляет $Q := \frac{1}{2n'}\sum\limits_{i=1}^{n'}\alpha_i$. С другой стороны, при наличии допустимого решения с качеством обслуживания не меньше этого Q соответствующее ему множество индексов $\mathcal{I}$ удовлетворяет условию задачи РАЗБИЕНИЕ. Отсюда следует NP-полнота задачи составления расписания работы операторов центра обработки вызовов в форме верификации свойств, а значит, и NP-трудность рассматриваемой максимизационной задачи. Теорема 1 доказана.

2. Генетический алгоритм

2.1. Общая схема алгоритма. Генетический алгоритм (ГА) представляет собой эвристический алгоритм оптимизации, в основу которого положены биологические принципы естественного отбора и изменчивости [12]. В процессе работы алгоритма происходит последовательная смена поколений, которые состоят из фиксированного числа особей — элементов пространства решений. Последовательность символов некоторого алфавита, кодирующая решение в популяции ГА, называется *генотипом*. В качестве меры приспособленности генотипа к условиям решаемой задачи рассматривается значение целевой функции для решения, соответствующего этому генотипу, с учётом «штрафа» в случае недопустимого решения. Более приспособленные особи в каждом следующем поколении в среднем получают больше потомков. Кроме того, в процессе формирования следующего поколения часть генотипов изменяется некоторым случайным образом в результате кроссинговера (скрещивания)

и мутации, а часть генотипов потомков полностью идентична родительским генотипам.

Обозначим множество всевозможных генотипов через B. Действие оператора селекции $\mathrm{Sel}\colon B^N \to \{1,\dots,N\}$ состоит в выборе номера родительской особи для построения очередного потомка, где N — число особей в популяции.

Действие оператора кроссинговера $\mathrm{Cross}\colon B \times B \to B \times B$ заключается в построении двух генотипов потомков на основе двух родительских генотипов. С вероятностью P_c каждый из потомков будет наследовать выбранные некоторым случайным образом гены родительских особей, а с вероятностью $1 - P_c$ потомки будут идентичны родителям.

Действие оператора мутации $\mathrm{Mut}\colon B \to B$ состоит в изменении поданного на вход генотипа. С вероятностью P_m каждый ген исходного генотипа изменяется случайным образом, а с вероятностью $1 - P_m$ остаётся без изменений.

Общая схема генетического алгоритма представлена ниже.

1: **for** $k = 1$ to N **do**
2: Построить случайным образом генотип $\xi^{k,0}$.
3: **for** $t = 0$ to $t_{\max}$ **do**
4: **for** $k = 1$ to $N/2$ **do**
5: Селекция: выбрать генотипы с учётом приспособленности особей $\xi \leftarrow \xi^{\mathrm{Sel}(\Pi^t),t}$, $\eta \leftarrow \xi^{\mathrm{Sel}(\Pi^t),t}$.
6: Скрещивание: построить $(\xi', \eta') \leftarrow \mathrm{Cross}(\xi, \eta)$.
7: Мутация: положить $\xi^{2k-1,t+1} \leftarrow \mathrm{Mut}(\xi')$, $\xi^{2k,t+1} \leftarrow \mathrm{Mut}(\eta')$.
8: Вычислить значение функции приспособленности для генотипов $\xi^{2k-1,t+1}$ и $\xi^{2k,t+1}$.
9: **return** лучшее из найденных решений по значению функции приспособленности за всё время работы алгоритма

2.2. Элитарная стратегия управления популяцией. В генетическом алгоритме при построении следующего поколения может сохраняться один или несколько элитных генотипов текущей популяции, имеющих высокую приспособленность. В реализации генетического алгоритма, описанном в п. 2.3, используется стратегия, в которой при построении следующего поколения сохраняются n' наиболее приспособленных генотипов.

2.3. Реализация генетического алгоритма. В этом пункте опишем предлагаемую реализацию генетического алгоритма, в котором для вычисления значения функции приспособленности используются следующие вспомогательные алгоритмы: алгоритм назначения операторов,

алгоритм сокращения дефицита операторов и алгоритм локальной корректировки решения. Далее будет приведено описание этих алгоритмов.

Генотип состоит из четырёх векторов: $\mathbf{a} = (a_1, \ldots, a_n)$, где a_p — время начала работы оператора $p \in P$, $\mathbf{b} = (b_1, \ldots, b_n)$, где b_p — время завершения работы оператора $p \in P$, $\mathbf{c} = (c_1, \ldots, c_n)$, где c_p — время начала перерыва оператора $p \in P$, $\mathbf{e} = (e_1, \ldots, e_n)$ — перестановка операторов для алгоритма назначения операторов (подробнее в п. 2.4).

В начальной популяции для каждой особи векторы $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ и $\mathbf{e}$ генерируются случайным образом с равномерным распределением с учётом ограничений на время работы и перерыва операторов, вектор $\mathbf{e}$ имеет вид $(1, 2, \ldots, n)$, где n — число операторов.

Пусть на каждой итерации все особи популяции упорядочиваются по возрастанию целевой функции. Номер особи в таком случае будем называть её *рангом*. В рассматриваемом генетическом алгоритме используется линейная ранговая селекция [13, 14], в которой вероятность выбора особи с рангом r равна $\frac{2r}{(N+1)N}$, $r = 1, \ldots, N$.

При кроссинговере для векторов $\mathbf{a}$, $\mathbf{b}$ и $\mathbf{c}$ применяется одноточечный оператор кроссинговера с единой для всех координатой скрещивания, которая выбирается случайным образом, а для вектора $\mathbf{e}$ применяется порядковый кроссинговер для задач на перестановках [15].

При мутации случайным образом выбирается оператор $p \in P$, затем для него случайным образом генерируются новые значения a_p, b_p и c_p с учётом ограничений на них. Для вектора $\mathbf{e}$ случайным образом выбираются индексы $i \in \{1, \ldots, n\}$ и $j \in \{1, \ldots, n\}$, где n — число операторов, затем элементы e_i и e_j меняются местами.

Решение $X = \{x_{pct}\}$, где $x_{pct} = 1$ тогда и только тогда, когда оператор p в интервал времени t назначен на обработку вызовов типа c, вычисляется на основе генотипа с помощью применения алгоритма назначения операторов, описанного в п. 2.4. Затем полученное решение улучшается с помощью алгоритмов сокращения дефицита операторов и локальной корректировки решения, которые описаны в п. 2.5 и 2.6 соответственно.

Значение функции приспособленности вычисляется по формуле

$$f(X) = \sum_{p \in P} \sum_{c \in C} \sum_{t \in T} s_{pc} x_{pct} - nkS(m(X) + u(X) + \max\{0, c(X) - F\}), \quad (17)$$

где $m(X)$ — общее число операторов, которых не хватает для обработки входящих вызовов согласно матрице X, $u(X)$ — число операторов $p \in P$, у которых число назначений на обработку вызовов менее H_p, $c(X)$ — итоговая стоимость оплаты труда операторов на основе матрицы X, F — объём фонда оплаты труда операторов, n — число операторов, k — число интервалов времени, S — максимальное значение навыка в обработке вызовов оператором.

Таким образом, функция приспособленности представляет собой целевую функцию за вычетом штрафов за нехватку операторов и за превышение фонда оплаты труда. С учётом штрафов значение функции приспособленности любого недопустимого решения будет меньше значения функции приспособленности любого допустимого решения.

Результатом работы алгоритма является матрица X, относящаяся к генотипу с наибольшим значением функции приспособленности среди всех генотипов, вычисленных за время выполнения алгоритма.

При отсутствии улучшений рекордного значения приспособленности на протяжении большого числа итераций происходит остановка ГА в соответствии со следующей известной схемой (см., например, [16–18]). Под улучшением рекордного значения приспособленности далее понимается увеличение приспособленности лучшего найденного генотипа в текущем поколении не менее чем на $\varepsilon\,\%$, где ε — настраиваемый параметр. Пусть t — текущее время выполнения алгорима в секундах, t_0 — момент времени, когда произошло последнее улучшение рекорда, k — номер текущего поколения. Алгорим завершает свою работу, если $t > T_{\max}$ или $t > 2t_0$, $k > D$ и найдено допустимое решение. Если $t > 2t_0$, $k > D$ и допустимое решение не найдено, то процесс эволюции инициализируется заново. Условие $t > 2t_0$ показывает отсутствие улучшения рекордного значения целевой функции за то же время, что было затрачено на получение этого рекорда. Условие $k > D$ гарантирует, что на начальных итерациях генетического алгоритма не будет преждевременно принято решение о перезапуске. Использование перезапусков ГА уменьшает вероятность завершения работы до получения допустимого решения. Применение того же правила для остановки ГА после получения допустимого решения позволяет существенно сократить время работы при незначительном снижении качества решения по сравнению с остановкой при достижении $t > T_{\max}$. В вычислительном эксперименте в разд. 3 значение $T_{\max}$ выбрано равным 30 минутам, $\varepsilon = 1\%$, $D = 100$.

2.4. Алгоритм назначения операторов. С помощью этого алгоритма производится заполнение трёхмерной булевой матрицы X, которая ставит в соответствие операторам типы вызовов, на обработку которых они назначены в различные интервалы времени. Работа алгоритма состоит из двух этапов. На каждом этапе производится назначение операторам типов вызовов, которые они будут обрабатывать в различные интервалы времени.

Цель первого этапа работы алгоритма состоит в том, чтобы по возможности полностью выполнить ограничения на минимальное число интервалов времени, в которые операторы должны быть назначены на обработку вызовов. Это достигается за счёт того, что если у некоторого оператора это ограничение выполнено, то предпочтение при назначении

на обработку вызовов отдаётся другому оператору, у которого оно ещё не выполнено.

На втором этапе производятся дополнительные назначения операторов на обработку вызовов с учётом ограничения на максимальное число интервалов времени, в которые оператор должен быть назначен на обработку вызовов.

На каждом этапе при попытке назначения оператору типа вызовов, которые он будет обрабатывать в данный интервал времени, учитываются график работы оператора (задаётся с помощью вектров $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$), типы вызовов, которые оператор может обрабатывать, его навыки в обработке вызовов различных типов и требуемое число операторов для обработки вызовов каждого типа в каждый интервал времени.

Пошаговое описание алгоритма приведено в приложении (алгорим 1).

2.5. Алгоритм сокращения дефицита операторов. Поскольку алгоритм назначения операторов «недальновиден», он не всегда может назначить достаточное число операторов для обработки вызовов, даже если это возможно. Алгоритм, рассматриваемый ниже, призван компенсировать данный недостаток. В ходе выполнения этого алгоритма дефицит операторов, назначенных на обработку вызовов некоторого типа, сокращается с помощью локальных корректировок матрицы X, если это возможно.

Для того типа вызова c, где есть дефицит операторов, производится поиск такого p' среди операторов, назначенных на обработку другого типа вызова, который может обработать данный тип вызова. Затем для него ищем замену p'' среди свободных операторов. Если такую пару (p', p'') удаётся найти, то производится переназначение и дефицит операторов уменьшается. Алгоритм работает до тех пор, пока существует дефицит операторов для обработки вызовов и пока существует пара операторов, переназначив которых, можно уменьшить дефицит.

Пошаговое описание алгоритма приведено в приложении (алгорим 2).

2.6. Алгоритм локальной корректировки решения. В некоторых случаях возможно увеличение значения функции приспособленности, если в матрице назначений пару операторов поменять местами. С целью отыскания таких пар просматриваются по порядку все интервалы времени t; для каждого фиксированного t просматриваются по порядку все операторы p'; для выбранного p' просматриваются по порядку все операторы p'' (начиная со следующего за p'). Если переназначение типов обрабатываемых вызовов между операторами p' и p'' в момент t возможно и приводит к увеличению приспособленности, то это переназначение производится и выполняется переход к следующему оператору p'.

Пошаговое описание алгоритма приведено в приложении (алгорим 3).

3. Вычислительный эксперимент

Тестирование проводилось на ЭВМ с процессором Intel(R) Xeon(R) X5675 3,07 ГГц и оперативной памятью 32 ГБ. Генетический алгоритм был реализован на языке C++. Для одной задачи параллельно запускалось четыре потока операционной системы, в каждом из которых выполнялся процесс эволюции независимо от остальных потоков. Данная схема выполнения эквивалентна четырём последовательным запускам генетического алгоритма. В качестве итогового решения выбиралось наилучшее решение по значению целевой функции среди всех запусков генетического алгоритма. Вероятности кроссинговера и мутации равны 0,4 и 0,25 соответственно, число особей в популяции равно 100, среди которых 15 элитных. Максимальное время работы ограничено 30 минутами, значение параметра ε равно 1%, а значение параметра D равно 100. Данные значения параметров были выбраны на основе предварительных экспериментов.

Для исследования работоспособности генетического алгоритма проводилось сравнение найденных им значений целевой функции с оптимальными значениями, найденными с использованием пакета CPLEX версии 20.1.0.1.

Для получения достаточного по объёму и разнообразию набора тестовых задач был реализован генератор задач, описание которого приведено в п. 3.1. Результаты тестирования представлены в п. 3.2.

3.1. Генерация данных. Для создания набора задач был создан генератор, пошаговое описание которого приведено в приложении.

Параметрами генератора являются число операторов n, число типов входящих вызовов m, число интервалов времени k, число интервалов времени на перерыв l, вероятность q наличия у оператора способности обрабатывать больше одного типа входящего вызова, максимальное число u типов входящих вызовов, которые способен обработать один оператор, вероятность v уменьшения требуемого числа операторов для каждого типа вызова в каждый интервал времени, коэффициент уменьшения w.

Тестовые задачи [1)] генерировались по наборам параметров $(n, m, v) = (64; 2; 0{,}5)$, $(64; 2; 0{,}25)$, $(64; 2; 0{,}1)$, $(128; 16; 0{,}5)$, $(128; 16; 0{,}25)$, $(128; 16; 0{,}1)$, $(256; 64; 0{,}5)$, $(256; 64; 0{,}25)$. Для остальных параметров были выбраны следующие значения: $t = 32$, $l = 4$, $q = 0{,}67$, $u = 5$. Для каждого фиксированного набора параметров генерировалось 30 задач, итого 240 задач.

[1)] Тестовые задачи и результаты доступны по ссылке `github.com/maximsakhno/call-center-scheduling-test-problems`

3.2. Результаты тестирования. Обозначим через r среднее отношение значения функции приспособленности, найденное генетическим алгоритмом, к значению целевой функции, найденному с помощью пакета CPLEX. В столбце r значения приведены только для тех серий, где пакетом CPLEX были найдены допустимые решения за отведённое время для всех примеров. Через R обозначим среднее отношение значения целевой функции, найденное генетическим алгоритмом, к оптимальному значению целевой функции или её верхней оценке, полученной пакетом CPLEX с использованием линейного программирования и отсечений. Усреднение проводится по серии задач, сгенерированных с фиксированным набором параметров. Отношение числа найденных допустимых решений за отведённое время к общему числу задач в серии для пакета CPLEX обозначим через D, а отношение числа оптимальных — через D^*. Отметим, что генетическим алгоритмом были найдены допустимые решения для всех тестовых задач. Среднее время работы генетического алгоритма в секундах обозначим через T_{GA}, а среднее время работы пакета CPLEX на тех задачах, для которых допустимое решение найдено за отведённое время, обозначим через T_{CPLEX}. Время работы пакета CPLEX было ограничено 1 часом.

Таблица 1

Результаты тестирования

n	m	v	D	D^*	r	R	T_{GA}, с	T_{CPLEX}, с
64	2	0,5	100%	80%	0,99	0,99	3,07	727,72
64	2	0,25	100%	80%	0,99	0,99	5,00	775,01
64	2	0,1	66%	33%	—	0,99	66,93	1867,39
128	16	0,5	100%	30%	0,93	0,93	123,20	2645,81
128	16	0,25	96%	16%	—	0,93	180,80	3200,98
128	16	0,1	43%	0%	—	0,93	333,93	3602,88
256	64	0,5	96%	0%	—	0,89	923,07	3616,34
256	64	0,25	0%	0%	—	0,89	914,80	—

Результаты вычислительного эксперимента представлены в табл. 1. Отметим, что пакетом CPLEX за отведённое время в трёх сериях найдены допустимые решения для всех примеров, в одной серии не найдено ни одного допустимого решения, а для остальных серий доля найденных допустимых решений варьируется от 43% до 96%. При этом нет ни одной серии примеров, для которой пакетом CPLEX за отведённое время найдены оптимальные решения для всех примеров. Для трёх из восьми серий пакетом CPLEX не найдено ни одного оптимального решения.

На тех сериях, для которых пакетом CPLEX найдены допустимые решения во всех примерах, значение r не меньше 0,93. На сериях примеров

с числом операторов, равным 64, среднее отклонение от оптимума для решений, полученных генетическим алгоритмом, не превосходит 1%, для 128 не превосходит 7%, а для 256 не превосходит 11%.

Также заметим, что для серий примеров с числом операторов n, равным 64, среднее время работы пакета CPLEX превосходило среднее время работы генетического алгоритма более чем в 25 раз, для $n = 128$ — более чем в 17 раз, а для $n = 256$ — более чем в 3 раза. Соотношение уменьшается с увеличением числа операторов в связи с тем, что среднее время работы пакета CPLEX расчитывается на основе тех задач, где пакетом найдены допустимые решения за отведённое время.

Таким образом генетический алгоритм находит допустимые решения для всех примеров, при этом среднее время работы генетического алгоритма по каждой серии не превосходит 16 минут, а отклонение от оптимума не превышает 11%. В то же время пакет CPLEX не всегда находит допустимые решения даже за 1 час, при этом время работы генетического алгоритма в среднем значительно меньше, чем у пакета CPLEX.

Заключение

В работе проведён анализ задачи составления расписания операторов центра обработки вызовов и разработан эффективный алгоритм её решения. Доказана NP-трудность поставленной задачи, и построена её математическая модель. Также предложен и протестирован генетический алгоритм её решения. Среднее отклонение от оптимума для решений, полученных генетическим алгоритмом, составило от 1 до 11%, что является приемлемым для его практического использования, поскольку при определении навыков, как правило, используются приближённые экспертные оценки. Для всех серий время работы генетического алгоритма было значительно меньше чем у пакета CPLEX.

Приложение

Генератор задач. Работа генератора состоит из следующих этапов.

1. Назначение для каждого оператора типов входящих вызовов, которые он может обработать. Для этого с вероятностью q определяется, может ли оператор обрабатывать более одного типа вызова. Если может, то случайным образом определяется число типов вызовов, которые он может обрабатывать в диапазоне от 2 до u. Затем для каждого оператора случайным образом выбираются конкретные типы входящих вызовов.

2. Назначение каждому оператору конкретного времени работы и перерыва случайным образом.

3. Назначение типа входящего вызова случайным образом для обработки в каждый интервал времени для каждого оператора (среди тех

Алгоритм 1. Алгоритм назначения операторов

1: Обнулить X.
2: **for** $J \in \{H, G\}$ **do**
3: **for** $t = 1$ to k **do**
4: **for** $i = 1$ to n **do**
5: $p := e_i$
6: **bool** condition1 := оператор p не работает в интервал времени t согласно векторам **a**, **b**, **c**;
condition2 := у оператора p число назначений не меньше чем J_p
7: **if** condition1 **or** condition2 **then continue**
8: Найти тип вызова c, при котором достигается наибольшее положительное значение $n_{ct} - \sum\limits_{p' \in P} x_{p'ct}$ среди тех вызовов, которые оператор p может обработать.
9: **if** такой тип не найден **then continue**
10: $x_{pct} := 1$

Алгоритм 2. Алгоритм сокращения дефицита операторов

1: **for** $t = 1$ to k **do**
2: **for** $c' = 1$ to m **do**
3: **if** $\sum\limits_{p \in P} x_{pc't} > n_{c't}$ **then continue**
4: **for** $p' = 1$ to n **do**
5: **if** $\sum\limits_{c \in C} x_{p'ct} = 0$ **or** $x_{p'c't} = 1$ **or** $m_{p'c'} = 0$ **then continue**
6: Выбрать c'' таким, что $x_{p'c''t} = 1$.
7: **for** $p'' = p' + 1$ to n **do**
8: **bool** condition := оператор p'' не работает в интервал времени t согласно векторам **a**, **b**, **c**
9: **if** condition **or** $\sum\limits_{c \in C} x_{p''ct} = 1$ **or** $m_{p''c''} = 0$ **then continue**
10: $x_{p'c't} := 1$; $x_{p'c''t} := 0$; $x_{p''c''t} := 1$
11: **break**
12: **break**

типов вызовов, которые он может обработать), если он в данный момент работает согласно расписанию.

4. Вычисление значения фонда оплаты труда для работы операторов согласно сгенерированному расписанию.

Алгоритм 3. Алгоритм локальной корректировки решения

for $t = 1$ to k **do**
  for $p' = 1$ to n **do**
    if $\sum\limits_{c \in C} x_{p'ct} = 0$ **then continue**
    Выбрать c' таким, что $x_{p'c't} = 1$.
    for $p'' = p' + 1$ to n **do**
      if $\sum\limits_{c \in C} x_{p''ct} = 0$ **then continue**
      Выбрать c'' таким, что $x_{p''c''t} = 1$.
      if $m_{p'c''} = 1$ **and** $m_{p''c'} = 1$ **and** $s_{p'c''} + s_{p''c'} > s_{p'c'} + s_{p''c''}$ **then** $x_{p'c't} := 0$; $x_{p''c''t} := 0$; $x_{p'c''t} := 1$; $x_{p''c't} := 1$

5. Вычисление требуемого числа операторов для обработки входящих вызовов каждого типа на основе построенного расписания.

6. Для каждого типа вызовов и интервала времени требуемое число операторов для обработки входящих вызовов умножается на w с вероятностью v.

ЛИТЕРАТУРА

1. **Gans N., Koole G., Mandelbaum A.** Telephone call centers: Tutorial, review, and research prospects // Manuf. Serv. Oper. Manag. 2003. V. 5, No. 2. P. 79–141.
2. **Avramidis A. N., Deslauriers A., L'Ecuyer. P.** Modeling daily arrivals to a telephone call center // Manag. Sci. 2004. V. 50. P. 896–908.
3. **Brown L., Gans N., Mandelbaum A., Sakov A., Shen H., Zeltyn S., Zhao L.** Statistical analysis of a telephone call center: A queueing-science perspective // J. Am. Stat. Assoc. 2010. V. 100. P. 36–50.
4. **Mehrotra V.** Ringing up big business // ORMS Today. 1997. V. 24. P. 18–24.
5. **Cezik M., L'Ecuyer P.** Staffing multi-skill call centers via linear programming and simulation // Manag. Sci. 2008. V. 54. P. 310–323.
6. **Avramidis A. N., Chan W., L'Ecuyer P.** Staffing multi-skill call centers via search methods and a performance approximation // IIE Trans. 2009. V. 41, No. 6. P. 483–497.
7. **Pot A., Bhulai S., Koole G.** A simple staffing method for multi-skill call centers // Manuf. Serv. Oper. Manag. 2008. V. 10, No. 3. P. 421–428.
8. **Zaozerskaya L. A.** A heuristic for a special case of the generalized assignment problem with additional conditions // J. Phys. Conf. Ser. 2021. V. 1791, No. 1, ID 012092. 6 p.
9. **Bhulai S., Koole G., Pot A.** Simple methods for shift scheduling in multi-skill call centers // Manuf. Serv. Oper. Manag. 2008. V. 10, No. 3. P. 411–420.

10. **Avramidis A. N., Chan W., Gendreau M., L'Ecuyer P., Pisacane O.** Optimizing daily agent scheduling in a multi-skill call center // Eur. J. Oper. Res. 2010. V. 200. P. 822–832.
11. **Гэри М., Джонсон Д.** Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 416 с.
12. **Holland J.** Adaptation in natural and artificial systems. Ann Arbor: Univ. Michigan Press, 1975. 183 p.
13. **Baker J. E.** Adaptive selection methods for genetic algorithms // Proc. Int. Conf. Genetic Algorithms and Their Applications (Pittsburgh, PA, USA, July 24–26, 1985). Pittsburgh, PA: Carnegie-Mellon Univ., 1985. P. 101–111.
14. **Lehre P. K.** Fitness-levels for non-elitist populations // Proc. 13th Annu. Conf. Genetic and Evolutionary Computation (Dublin, Ireland, July 12–16, 2011). New York: ACM, 2011. P. 2075–2082.
15. **Borisovsky P., Dolgui A., Eremeev A.** Genetic algorithms for a supply management problem: MIP-recombination vs greedy decoder // Eur. J. Oper. Res. 2009. V. 195. P. 770–779.
16. **Hampson S., Kibler D.** Large plateaus and plateau search in Boolean Satisfiability problems: When to give up searching and start again // Cliques, Coloring and Satisfiability. Proc. 2nd DIMACS Implementation Challenge, Workshop (Piscataway, USA, Oct. 11–13, 1993). Providence: AMS, 1996. P. 437–456.
17. **Balas E., Niehaus W.** Optimized crossover-based genetic algorithms for the maximum cardinality and maximum weight clique problems // J. Heuristics. 1998. V. 4, No. 2. P. 107–122.
18. **Тюнин Н. Н., Еремеев А. В.** Алгоритм дифференциальной эволюции для оптимизации направленности фазированных антенных решёток // Мат. структуры и моделирование. 2022. № 3. С. 57–68.

Еремеев Антон Валентинович
Сахно Максим Алексеевич

Статья поступила
29 апреля 2022 г.
После доработки —
25 января 2023 г.
Принята к публикации
13 февраля 2023 г.

DISKRETNYI ANALIZ I ISSLEDOVANIE OPERATSII
/DISCRETE ANALYSIS AND OPERATIONS RESEARCH/
April–June 2023. Vol. 30, No. 2. P. 48–66

UDC 519.85 DOI 10.33048/daio.2023.30.735

THE PROBLEM OF SCHEDULING FOR CALL-CENTRE OPERATORS

A. V. Eremeev[a] and *M. A. Sakhno*[b]

Sobolev Institute of Mathematics, Omsk Department,
13 Pevtsov Street, 644099 Omsk, Russia
E-mail: [a] eremeev@ofim.oscsbras.ru, [b] maxim.sakhno@gmail.com

Abstract. The paper is devoted to solving the problem of scheduling the work of the call center staff. A model of integer linear programming is formulated, the problem is shown to be NP-hard, and a genetic algorithm is proposed that takes into account the specifics of the problem. An experimental comparison of optimal solutions obtained using the CPLEX package with solutions found by the genetic algorithm is carried out. The computational experiment showed the practically acceptable accuracy of the solutions obtained by the genetic algorithm and its applicability to large-dimensional problems. Tab. 1, bibliogr. 18.

Keywords: call center, integer linear programming, genetic algorithm, computational complexity.

REFERENCES

1. **N. Gans, G. Koole,** and **A. Mandelbaum,** Telephone call centers: Tutorial, review, and research prospects, *Manuf. Serv. Oper. Manag.* **5** (2), 79–141 (2003).
2. **A. N. Avramidis, A. Deslauriers,** and **P. L'Ecuyer,** Modeling daily arrivals to a telephone call center, *Manag. Sci.* **50**, 896–908 (2004).
3. **L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn,** and **L. Zhao,** Statistical analysis of a telephone call center: A queueing-science perspective, *J. Am. Stat. Assoc.* **100**, 36–50 (2010).
4. **V. Mehrotra,** Ringing up big business, *ORMS Today* **24**, 18–24 (1997).
5. **M. Cezik** and **P. L'Ecuyer,** Staffing multi-skill call centers via linear programming and simulation, *Manag. Sci.* **54**, 310–323 (2008).

This research is supported by the Russian Science Foundation (Project 21–41–09017).

English version: Journal of Applied and Industrial Mathematics **17** (2) (2023).

6. **A. N. Avramidis, W. Chan,** and **P. L'Ecuyer,** Staffing multi-skill call centers via search methods and a performance approximation, *IIE Trans.* **41** (6), 483–497 (2009).
7. **A. Pot, S. Bhulai,** and **G. Koole,** A simple staffing method for multi-skill call centers, *Manuf. Serv. Oper. Manag.* **10** (3), 421–428 (2008).
8. **L. A. Zaozerskaya,** A heuristic for a special case of the generalized assignment problem with additional conditions, *J. Phys. Conf. Ser.* **1791** (1), ID 012092, 6 p. (2021).
9. **S. Bhulai, G. Koole,** and **A. Pot,** Simple methods for shift scheduling in multi-skill call centers, *Manuf. Serv. Oper. Manag.* **10** (3), 411–420 (2008).
10. **A. N. Avramidis, W. Chan, M. Gendreau, P. L'Ecuyer,** and **O. Pisacane,** Optimizing daily agent scheduling in a multi-skill call center, *Eur. J. Oper. Res.* **200**, 822–832 (2010).
11. **M. R. Garey** and **D. S. Johnson,** *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, 1979; Mir, Moscow, 1982 [Russian]).
12. **J. Holland,** *Adaptation in Natural and Artificial Systems* (Univ. Michigan Press, Ann Arbor, 1975).
13. **J. E. Baker,** Adaptive selection methods for genetic algorithms, in *Proc. Int. Conf. Genetic Algorithms and Their Applications, Pittsburgh, PA, USA, July 24–26, 1985* (Carnegie-Mellon Univ., Pittsburgh, PA, 1985), pp. 101–111.
14. **P. K. Lehre,** Fitness-levels for non-elitist populations, in *Proc. 13th Annu. Conf. Genetic and Evolutionary Computation, Dublin, Ireland, July 12–16, 2011* (ACM, New York, 2011), pp. 2075–2082.
15. **P. Borisovsky, A. Dolgui,** and **A. Eremeev,** Genetic algorithms for a supply management problem: MIP-recombination vs greedy decoder, *Eur. J. Oper. Res.* **195**, 770–779 (2009).
16. **S. Hampson** and **D. Kibler,** Large plateaus and plateau search in Boolean Satisfiability problems: When to give up searching and start again, in *Cliques, Coloring and Satisfiability* (Proc. 2nd DIMACS Implementation Challenge, Workshop, Piscataway, USA, Oct. 11–13, 1993) (AMS, Providence, 1996), pp. 437–456.
17. **E. Balas** and **W. Niehaus,** Optimized crossover-based genetic algorithms for the maximum cardinality and maximum weight clique problems, *J. Heuristics* **4** (2), 107–122 (1998).
18. **N. N. Tyunin** and **A. V. Eremeev,** Differential evolution for directivity optimization of short-wave phased antenna arrays, *Mat. Strukt. Model.,* No. 3, 57–68 (2022) [Russian].

Anton V. Eremeev
Maxim A. Sakhno

Received April 29, 2022
Revised January 25, 2023
Accepted February 13, 2023