УДК 681.3.06

МОДЕЛЬ ЗАДАЧИ НАХОЖДЕНИЯ ОПТИМАЛЬНОГО ПЛАНА РАБОТ И ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ДЛЯ ЕЕ РЕШЕНИЯ

М. Б. Родькина, Т. М. Леденева

Воронежский государственный университет

Поступила в редакцию 07.04.2011 г.

Аннотация. В статье представлена модель для задачи составления оптимального плана работ на примере отдела информационных технологий, сотрудники которого работают с запросами на выполнение определённых операций, поступающими извне. Для решения задачи используется генетический алгоритм, в котором каждая особь наделена способностью влиять на динамику своей популяции и, в конечном итоге, на реализацию и характеристики алгоритма в целом.
Ключевые слова: расписание, генетический алгоритм, особь, установка.
Annotation: The mathematical model of problem of scheduling for department of the information technology is considered in the paper. Employees works on requests to perform certain actions, that are coming from the outside. Genetic algorithm is used to solve this problem. Each individual of algorithm has an ability to influence it's population evolution and behavior of algorithm as a whole.
Key words: schedule, genetic algorithm, individual, guideline.

ВВЕДЕНИЕ

В настоящее время практически в каждой крупной организации есть отдел информационных технологий (IT-отдел), к функциям которого относятся информационная и техническая поддержка процессов принятия решений. Деятельность сотрудников отдела связана с обработкой запросов на выполнение определённых операций. Запросы поступают извне, носят разнородный характер, и в течение фиксированного промежутка времени их количество может существенным образом меняться. Однако, запросы, ожидающие выполнения на конкретный момент времени, можно объединить в статическую группу. Для экономии времени и повышения эффективности работы можно оптимально упорядочить запросы внутри этой группы. Целью данной статьи является представление модели планирования деятельности на примере IT-отдела и генетического метода, позволяющего найти субоптимальное решение данной задачи за приемлемое время.

1. ПОСТАНОВКА ЗАДАЧИ И ОПТИМИЗАЦИОННАЯ МОДЕЛЬ СОСТАВЛЕНИЯ РАСПИСАНИЯ С УЧЕТОМ ИНТЕРВАЛЬНЫХ ОЦЕНОК ВРЕМЕНИ ВЫПОЛНЕНИЯ РАБОТ

Рассмотрим положения, определяющие условия функционирования IT-отдела.

1. В отделе работает m сотрудников. Все запросы, ожидающие решения на данный момент времени (n), известны всем сотрудникам. Любой сотрудник может принять на исполнение любой ожидающий запрос.

2. Известно время t_{ij} выполнения каждого запроса j каждым сотрудником i, т.е. задана таблица T вида

Таблица 1

Время выполнения запросов сотрудниками

	1	2	...	n
1	t_{11}	t_{12}	...	t_{1n}
2	t_{21}	t_{22}	...	t_{2n}
...	...	...	...	...
m	t_{m1}	t_{m2}	...	t_{mn}

3. Запрос может выполняться в одну операцию (простой запрос), а может потребовать нескольких операций для своего выполнения (сложный запрос). Количество операций запроса i обозначим через k_i.

4. Операции и простых, и сложных запросов выполняются без прерываний.

5. Некоторые операции запросов могут параллельно выполняться несколькими сотрудниками. Запрос может содержать операции, которые могут выполняться одновременно одним сотрудником.

6. Каждый запрос i имеет директивный срок d_i – срок, к которому запрос должен быть исполнен.

7. Сотрудник может бездействовать только в том случае, когда ожидающих запросов нет.

Необходимо назначить и упорядочить запросы для всех сотрудников так, чтобы время выполнения всех запросов было минимальным, и максимальное количество запросов было исполнено без превышения их директивного срока.

Минимизация общего времени исполнения поступающих запросов возможна за счёт:

1) минимизации времени выполнения каждого запроса i путём назначения его такому сотруднику j, который выполнит его за минимальное время, т. е. $t_{ji} = \min\limits_{k=\overline{1,m}} t_{ki}$;

2) параллельной обработки несколькими сотрудниками тех операций запросов, которые это допускают;

3) параллельной обработки сотрудниками нескольких операций;

4) для каждого запроса минимизации разности между временем выполнения запроса и его директивным сроком $(f_i - d_i)$, $i = \overline{1,m}$, где f_i – время завершения запроса i.

В рамках перечисленных положений задача заключается в определении расписания, которое минимизирует общее время выполнения запросов.

Пусть $x_{ijk} = 1$, если операция i выполняется сотрудником k j-й по порядку, $x_{ijk} = 0$, в противном случае; f_i – время завершения всех операций запроса i; D – время завершения выполнения всех n запросов; $S = \sum\limits_{i=1}^{n} k_i$ – общее количество операций во всех запросах. Пусть величина μ_i показывает, превысило ли время завершения запроса i директивный срок d_i, если для конкретного решения $\mu_i = 0$, то запрос i исполняется вовремя, иначе, запрос будет исполнен с превышением директивного срока.

Модель задачи в общем случае выглядит следующим образом:

$$\sum_{i=1}^{n} \mu_i \to \min, \tag{1}$$

$$D \to \min, \tag{2}$$

$$\mu_i = \begin{cases} 0, & f_i - d_i \le 0 \\ 1, & f_i - d_i > 0 \end{cases}, \; i = \overline{1,n}, \tag{3}$$

$$1 \le \sum_{k=1}^{m} \sum_{j=1}^{S} x_{ijk} \le m, \; i = \overline{1,S}, \tag{4}$$

$$\begin{gathered} \sum_{k=1}^{m} \sum_{i=1}^{S} x_{ijk} \le m, \; j = \overline{1,S}, \\ x_{ijk} \in \{0,1\}. \end{gathered} \tag{5}$$

Время считается дискретным. Если две операции i_1 и i_2 выполняются параллельно сотрудником k, то полагается, что сотрудник занят ими на протяжении времени $t = \max\left(t_{ki_1}, t_{ki_2}\right)$. Если два сотрудника k_1 и k_2 параллельно выполняют одну операцию i, то каждый из них за единицу времени выполняет соответственно часть $\frac{1}{t_{k_1 i}}$ и $\frac{1}{t_{k_2 i}}$ этой операции. Если в текущую единицу времени у одного из сотрудников (например, k_1) есть возможность полностью завершить операцию (то есть не выполненная часть $t_{\text{ост}} \le \frac{1}{t_{k_1 i}}$), то второй сотрудник освобождается от её выполнения. Иначе, оба сотрудника затрачивают ещё одну единицу времени, даже, если $t_{\text{ост}} < \frac{1}{t_{k_1 i}} + \frac{1}{t_{k_2 i}}$.

Заметим, что в реальной ситуации сотрудник, как правило, затрудняется указать точное время выполнения работы, но в состоянии дать оптимистическую и пессимистическую оценку этого времени. Исходя из этого, будем считать, что известны интервальные переменные $\tilde{t}_{ij} = \left[\underline{t_{ij}}, \overline{t_{ij}}\right]$. Тогда время завершения запроса $\tilde{f}_i = \left[\underline{f_i}, \overline{f_i}\right]$ для $i = \overline{1,n}$ также можно считать интервальной переменной. Так как для любого запроса i директивный срок запроса d_i – это число, а время завершения запроса задано интервалом $\left[\underline{f_i}, \overline{f_i}\right]$, то в зависимости от их взаимного расположения на числовой прямой, можно судить о вероятности, с которой запрос будет исполнен до директивного срока. Возможны следующие три случая (рис. 1).

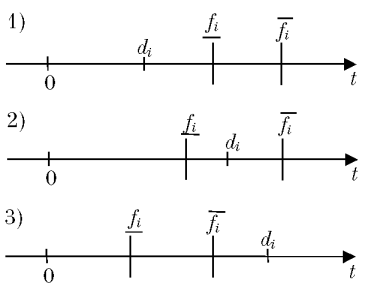

Рис. 1. Варианты взаимного расположения на оси времени директивного срока и времени завершения запроса

Как видно из рисунка, в первом случае директивный срок превышается с вероятностью $\mu_i = 1$. Во втором случае, вероятность превышения срока вычисляется по формуле $\mu_i = \frac{d_i - \underline{f_i} + 1}{\overline{f_i} - \underline{f_i} + 1}$. В последнем случае запрос всегда будет исполняться за время в пределах директивного срока, то есть $\mu_i = 0$.

Таким образом, для $i = \overline{1,n}$

$$\mu_i = \begin{cases} 0, & d_i \geq \overline{f_i}, \\ \dfrac{d_i - \underline{f_i} + 1}{\overline{f_i} - \underline{f_i} + 1}, & \underline{f_i} \leq d_i < \overline{f_i}, \\ 1, & d_i < \underline{f_i}. \end{cases} \qquad (6)$$

Тогда каждому расписанию можно поставить в соответствие вектор вероятностей $\mu = (\mu_1, ..., \mu_n)$. Чем меньше вероятность μ_i для каждого запроса i, тем лучше расписание с точки зрения его выполнимости.

Введём правила сравнения двух векторов μ^1 и μ^2.

1. Пусть μ_0^1 и μ_0^2 – число нулей в векторах μ^1 и μ^2 соответственно. Тогда

а) если $\mu_0^1 > \mu_0^2$, то будем считать, что μ^1 лучше μ^2, так как при расписании с вектором μ^1 большее количество запросов точно будут исполнены вовремя, чем при расписании с вектором μ^2. Обозначим эту ситуацию как $\mu^1 \succ \mu^2$;

b) если $\mu_0^1 < \mu_0^2$, тогда μ^1 хуже μ^2, и обозначим это как $\mu^1 \prec \mu^2$;

c) если $\mu_0^1 = \mu_0^2$, будем считать, что μ^1 и μ^2 эквивалентны по первому правилу, и для этого случая введём обозначение $\mu^1 \overset{1}{\approx} \mu^2$.

2. Пусть векторы μ^1 и μ^2 эквивалентны по первому правилу, то есть $\mu^1 \overset{1}{\approx} \mu^2$, и пусть c_1 – количество индексов i, для которых $\mu^1{}_i > \mu^2{}_i$, c_2 – тех, для которых $\mu^1{}_i < \mu^2{}_i$. Тогда

а) если $c_2 > c_1$, то будем считать, что μ^1 лучше μ^2 ($\mu^1 \succ \mu^2$), так как при расписании с вектором μ^1, большее количество запросов с большей вероятностью будут исполнены без превышения своих директивных сроков, чем при расписании с вектором μ^2;

b) если $c_2 < c_1$, то μ^1 хуже μ^2, то есть $\mu^1 \prec \mu^2$;

c) если $c_1 = c_2 = c$, положим, что μ^1 и μ^2 эквивалентны по второму правилу и обозначим эту ситуацию как $\mu^1 \overset{2}{\approx} \mu^2$.

3. Пусть векторы μ^1 и μ^2 эквивалентны по первому и второму правилам, то есть $\mu^1 \overset{1}{\approx} \mu^2$ и $\mu^1 \overset{2}{\approx} \mu^2$. Введём величины $\delta_1 = \frac{1}{c} \sum_{i:\mu_i^1 > \mu_i^2} (\mu_i^1 - \mu_i^2)$ и $\delta_2 = \frac{1}{c} \sum_{i:\mu_i^1 < \mu_i^2} (\mu_i^2 - \mu_i^1)$. δ_1 – фактически, среднее отклонение вектора μ^1 от вектора μ^2 по запросам с большей вероятностью превышения директивных сроков, показывает насколько в среднем μ^1 хуже μ^2. Аналогично δ_2 – среднее отклонение μ^2 от μ^1. Тогда

а) если $\delta_2 > \delta_1$, то μ^1 лучше μ^2, то есть $\mu^1 \succ \mu^2$;

b) если $\delta_2 < \delta_1$, то μ^1 хуже μ^2, что обозначается как $\mu^1 \prec \mu^2$;

c) если $\delta_1 = \delta_2$, то будем считать, что μ^1 и μ^2 эквивалентны.

Для решения задачи нам необходимо оптимизировать интервальную длину расписания D. Для простоты будем считать, что оценки времени выполнения запросов заданы так, что наиболее вероятное реальное значение – это середина интервала.

Будем сравнивать интервальные переменные $\tilde{f}_1 = \left[\underline{f_1}, \overline{f_1}\right]$ и $\tilde{f}_2 = \left[\underline{f_2}, \overline{f_2}\right]$ по правилам.

1. Пусть $f_1^{\text{ср}} = \frac{\underline{f_1} - \overline{f_1}}{2}$ и $f_2^{\text{ср}} = \frac{\underline{f_2} - \overline{f_1}}{2}$ – середины интервалов $\tilde{f}_1$ и $\tilde{f}_2$ соответственно. Тогда

а) если $f_1^{\text{ср}} < f_2^{\text{ср}}$, то будем считать, что $\tilde{f}_1$ лучше $\tilde{f}_2$ ($\tilde{f}_1 \succ \tilde{f}_2$);

b) если $f_1^{\text{ср}} > f_2^{\text{ср}}$, тогда положим, что $\tilde{f}_1$ хуже $\tilde{f}_2$, то есть $f_1^{\text{ср}} \prec f_2^{\text{ср}}$;

c) если $f_1^{\text{ср}} = f_2^{\text{ср}}$, то $\tilde{f}_1$ и $\tilde{f}_2$ эквивалентны по среднему. Обозначим такую эквивалентность $\tilde{f}_1 \overset{\text{ср}}{\approx} \tilde{f}_2$.

Данное правило обусловлено тем, что, исходя из условия (интервалов $\tilde{t}_{ij} = \left[\underline{t_{ij}}, \overline{t_{ij}}\right]$) и следуя правилам интервальной арифметики в процессе вычисления длины расписания, середина любого интервала всегда будет наиболее вероятным реальным значением, поэтому это значение можно рассматривать в качестве приоритета при сравнении.

2. Пусть, интервалы $\tilde{f}_1$ и $\tilde{f}_2$ эквивалентны по среднему, то есть $\tilde{f}_1 \overset{\text{ср}}{\approx} \tilde{f}_2$, для каждого интервала рассмотрим расстояние от середины до конца, $f_1^{\text{отк}} = \overline{f_1} - f_1^{\text{ср}}$ и $f_2^{\text{отк}} = \overline{f_2} - f_2^{\text{ср}}$. Тогда

а) если $f_1^{\text{отк}} < f_2^{\text{отк}}$, то будем считать, что $\tilde{f}_1$ лучше $\tilde{f}_2$ ($\tilde{f}_1 \succ \tilde{f}_2$).Это правило отражает пессимистический настрой и отсутствие склонности к риску, так как, реальное значение в своём

крайнем состоянии может находиться и на левом, и на правом концах интервала. При пессимистическом настрое мы жертвуем возможным лучшим значением на левом конце интервала, чтобы избежать возможности получить худшее значение на правом конце;

b) если $f_1^{\text{отк}} > f_2^{\text{отк}}$, то соответственно $\tilde{f}_1$ хуже $\tilde{f}_2$ ($\tilde{f}_1 \prec \tilde{f}_2$);

c) если $f_1^{\text{отк}} = f_2^{\text{отк}}$, то интервалы можно считать эквивалентными ($\tilde{f}_1 \approx \tilde{f}_2$).

Таким образом, получаем модель задачи:

$$\mu \to \max, \tag{7}$$

$$D \to \max, \tag{8}$$

$$1 \le \sum_{k=1}^{m} \sum_{j=1}^{S} x_{ijk} \le m,\ i = \overline{1,S}, \tag{9}$$

$$\sum_{k=1}^{m} \sum_{i=1}^{S} x_{ijk} \le m,\ j = \overline{1,S}, \quad x_{ijk} \in \{0,1\}. \tag{10}$$

Здесь максимальное значение каждого критерия – это лучшее его значение в соответствии с правилами сравнения, представленными выше. Данная задача относится к задачам теории расписаний, а, следовательно, является задачей дискретной оптимизации. Большинство задач этого класса являются NP-полными. В связи с этим для их решения, особенно при больших размерностях, актуальны приближённые методы, основанные на различных эвристических соображениях и позволяющие получить субоптимальное решение за приемлемое время.

2. РЕАЛИЗАЦИЯ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

Генетические алгоритмы (ГА) – это стохастические, эвристические оптимизационные методы, цель которых заключается в том, чтобы найти лучшее возможное, но не гарантированно оптимальное решение задачи. Основная структура данных алгоритма – это особь, заданная набором хромосом, при этом каждая хромосома, как правило, представляет собой битовую строку. Операторы кроссинговера и мутации позволяют получать новые решения задачи и сохраняют допустимость генотипов потомков. Вероятностный отбор призван обеспечить получение потомков с лучшим значением целевой функции от хорошо приспособленных родителей [1].

Базовый генетический алгоритм имеет следующую схему [2]:

Инициализация: создание начальной популяции.

Цикл по поколениям, пока не выполнено условие останова:

оценка приспособленности каждой особи популяции;

отбор родителей по приспособленности;

скрещивание родительских особей и получение потомков;

мутация потомков;

формирование популяции следующего поколения.

Конец цикла по выполнению критерия останова.

Генетические алгоритмы практически не поддаются настройке, которая могла бы увеличить их эффективность. Можно усложнить стандартный алгоритм, наделив каждую особь возможностью принятия решений и влияния на жизнь популяции. Тогда, основываясь на влиянии таких решений на поведение алгоритма, можно попробовать осуществить его тонкую настройку. Тогда алгоритм будет в процессе работы адаптироваться под специфику конкретной задачи

Предлагаемый алгоритм оперирует несколькими популяциями, начиная с одной. В процессе работы число популяции может изменяться.

По сравнению с базовым генетическим алгоритмом нами расширен набор параметров, характеризующих особь. В общем случае особь определяется набором параметров: генотип; пол; возраст; установка; вероятность участия в скрещивании; склонность к миграции; приспособленность и качества лидера.

Генотип – m хромосом, заданных перестановками $\sigma_i = (i_1, \ldots i_{n_i})$, $i = \overline{1,m}$, где n_i – число операций, выполняемых сотрудником i. Каждая перестановка σ_i однозначно определяет порядок выполнения операций сотрудником i.

Особь имеет пол: мужской или женский. В скрещивании участвуют особи разного пола. Мужской особью считается та, у которой больше нечётных хромосом, чем чётных. Если m чётно, и количества чётных и нечётных хромосом равны, то рассматриваются первые $k = m/2$ хромосом, и пол определяется по соотношению чётных и нечётных перестановок в k. Если равенство чётных и нечётных хромосом сохраняется, то число k снова уменьшает-

ся в 2 раза и т. д. При $k = 2$ пол определяет первая хромосома.

В качестве функции приспособленности в алгоритме используется функция $\varphi(\sigma) = \tilde{f} \cdot \left(1 + \frac{1}{S}\sum_{i=1}^{S} \mu_i\right)$, где σ – расписание, определяющее генотип особи, $\tilde{f} = \left[\underline{f}, \overline{f}\right]$ – интервал длины этого расписания. Чем хуже значение функции приспособленности, тем более приспособленной считается особь.

У каждой особи есть такая характеристика как возраст. Возраст каждой новой особи равен 0, и он увеличивается на 1 на каждой итерации алгоритма.

На каждом шаге алгоритма каждая особь решает, будет ли она участвовать в скрещивании. Чем меньше возраст особи и выше её приспособленность, тем больше вероятность участия. Эта вероятность также возрастает, если популяция начинает вырождаться, то есть её численность уменьшается на протяжении нескольких итераций алгоритма.

После того, как определены две группы особей (мужских и женских), принимающих участие в кроссовере, среди них проводится рулеточный отбор, пока меньшая по численности из групп не закончится. Из каждой группы выбирается особь для кроссовера. Вероятность выбора особи i на каждом этапе отборе задаётся формулой: $P(i) = \frac{\varphi_i}{\sum_{j \in G} \varphi_j}$, где G – гендерная группа, к которой относится особь. В результате скрещивания может получиться от одного до четырёх потомков. Все дочерние особи включаются в ту популяцию, к которой относятся родители. Для получения потомков используется упорядоченный кроссовер.

Если генотип потомка совпадает с генотипом уже существующей особи, то потомок подвергается мутации. При мутации все хромосомы особи собираются в одну, и случайно переставляется какая-либо граница между хромосомами, либо производится транспозиция выбранных случайным образом элементов.

Чем больше возраст особи и ниже её приспособленность, тем выше вероятность её гибели. Эта вероятность для особи i определяется по формуле:

$$P_{death} = \left(1 - \frac{1}{v_i}\right) \cdot \left(1 - \frac{\varphi_i}{\varphi_{\max}}\right) \times \left(1 - \eta_i\right) \cdot \min\left\{\frac{P}{G_{\max}}, 1\right\}.$$

Здесь P – число особей популяции, $G_{\max}$ – параметр алгоритма (число особей, больше которого нежелательно иметь популяции), η_i – установка особи.

Каждая особь популяции i имеет установку $\eta_i \in [0,1]$ на: гуманное, нейтральное или агрессивное поведение. Чем больше η_i, тем агрессивнее особь. Установка определяется при рождении, зависит от установки родителей и окружения и не меняется на протяжении жизни особи.

Чем приспособленнее особь, тем больший вес имеет её мнение при принятии коллективных решений. Особь с лучшим значением функции приспособленности по популяции считается лидером в этой популяции.

Популяция может находиться в трех состояниях: состоянии стабильности, упадка и процветания. Состояние определяется соотношением численности популяции и порогами упадка ($\underline{T}$) и процветания ($\overline{T}$). Популяция находится в стабильном состоянии, если количество её особей принадлежит отрезку $[\underline{T}, \overline{T}]$, и количество мужских и женских особей примерно одинаково.

Если численность популяции становится меньше порога упадка, или число особей одного пола превышает количество особей противоположного пола более чем в 1,5 раза, то популяция переходит в состояние упадка. Если численность популяции превышает порог процветания, то популяция переходит в состояние процветания независимо от соотношения полов в ней.

В состоянии упадка популяция может объявить войну другой популяции с целью захвата новых особей, некоторое время искусственно изменять потомков (тогда возрастает вероятность получения большего числа потомков при скрещивании, либо мутация направлена на выравнивание соотношения полов) или принять решение о вырождении (тогда все особи мигрируют в другие популяции, и популяция исчезает).

Чем больше популяция процветает, тем выше возможность миграции её особей и вероятность того, что часть популяции отделится вместе с каким-нибудь молодым лидером и образует новую популяцию.

Останов алгоритма происходит по количеству скрещиваний. Другие остановы не используются.

3. ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ

Для решения задачи было разработано программное приложение, реализующее предложенный ГА. Решения, полученные алгоритмом, сравнивались с точными решениями примеров. Тестирование проводилось на примерах различной размерности (максимальное число операций –500, сотрудников – 50) с разными параметрами алгоритма: возрастной порог, пороги процветания и упадка, максимальное число особей популяции, параметр, влияющий на вероятность принятия особью участия в скрещивании и др.

В результате тестирования было выявлено, что при прочих равных начальных параметрах больше всего на результат влияет число итераций алгоритма. Однако есть такое число итераций, что последующие его увеличения не дают улучшения результата. Например, при количестве сотрудников, равном 20, и числе операций – 40, необходимо около 15000 итераций (рис.2).

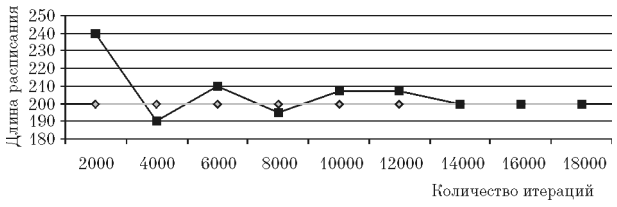

Рис. 2. Зависимость решения от количества операций

Решение задачи в общем случае может быть как больше, так и меньше оптимального из-за того, что расписание, лучшее по длине, не всегда удовлетворяет директивным срокам запросов.

В ходе тестирования также было выявлено, что размерность задачи мало влияет на максимальное число популяций. При увеличении числа итераций число популяций сначала стабильно равно единице, потом оно начинает возрастать, но очень быстро стабилизируется снова (рис. 3).

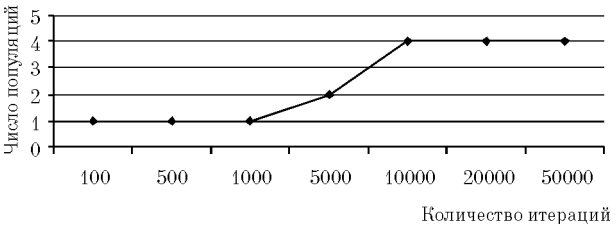

Рис. 3 Зависимость максимального числа популяций от количества итераций

На максимальное число итераций влияют пороги процветания и упадка, если определить ими малые популяции, то максимальное число популяций будет возрастать, иначе – уменьшаться.

Тестирование в целом показало работоспособность и эффективность алгоритма при решении поставленной задачи.

ЗАКЛЮЧЕНИЕ

При настоящих темпах передачи информации и компьютеризации многих сфер профессиональной деятельности работа IT-отдела оказывает существенное влияние на производительность всей организации. Грамотное распределение поручений между сотрудниками на основе предложенной модели планирования и генетического алгоритма позволят сделать работу IT-отдела более эффективной, что, в конечном счёте, может отразиться на прибыли организации.

СПИСОК ЛИТЕРАТУРЫ

1. *Богатырёв М. Ю.* Генетические алгоритмы: принципы работы, моделирование, применение / М. Ю. Богатырёв – Тула : ТГУ, 2003. – 152 с.

2. *Каширина И. Л.* Введение в эволюционное моделирование: учеб. пособие / И. Л. Каширина. – Воронеж : ВГУ, 2007. – 36 с.

Родькина М. Б. – аспирант кафедры вычислительной математики факультета Прикладной математики, механики и информатики Воронежского государственного университета. E-mail: llien@list.ru.

Rodkina M.B. – Post-graduate student, The dept. of the Mathematical Methods of Oreration Research, Voronezh State University. Tel. (4732) 208-282. E-mail: llien@list.ru.

Леденева Т. М. – заведующий кафедрой вычислительной математики факультета Прикладной математики, механики и информатики Воронежского государственного университета, доктор технических наук, профессор. Тел. (4732) 208-282. E-mail: dean@amm.vsu.ru

Ledeneva Tatyna Michaylovna – Doctor of Technic Sciences, Professor, The dept. of the Mathematical Methods of Oreration Research, Voronezh State University. Tel. (4732) 208-282. E-mail: dean@amm.vsu.ru