© 2012 г. И. А. БЫЧКОВ
(Московский государственный университет имени М. В. Ломоносова,
факультет вычислительной математики и кибернетики)

АЛГОРИТМ ВЫБОРА ПОРЯДКА АБОНЕНТОВ В СЕТИ С ТОПОЛОГИЕЙ КОЛЬЦА С АРБИТРАЖЕМ

В работе выполнена постановка задачи выбора порядка абонентов в кольце с арбитражем и предложен генетический алгоритм её решения. Осуществлён подбор значений параметров алгоритма, для которых алгоритм показывает хорошие результаты. Выполнено сравнение времени поиска порядка абонентов с построение полного расписания для разработанного генетического алгоритма и алгоритма случайного поиска. В результате сравнения установлено, что на исследованных данных генетический алгоритм находит порядок абонентов с построением полного расписания в 1,7 раз быстрее чем алгоритм случайного поиска.

AN ALGORITHM FOR SELECTING NODE ORDER IN THE NETWORK WITH ARBITRATED LOOP TOPOLOGY

/ I. A. Bychkov (Lomonosov Moscow State University, Faculty of Computational Mathematics and Cybernetics, Russia, 119991, Moscow, GSP-1, 1-52, Leninskiye Gory, E-mail: ibychkov@lvk.cs.msu.su)

The issue of this paper is the problem of selecting node orded in the network with arbitrated loop topology. The formal statement of the concerned problem is posed. A genetic algorithm for solving the posed problem is suggested. The values of the algorithm's parameters, on which it exhibits good results, are searched. The time need to find node order with building full schedule on it for the proposed genetic algorithm and a random search algorithm are compared. It has been established that the genetic algorithm finds node order with building full schedule on it 1.7 times faster than the random search algorithm.

1. Введение

В данной работе рассматриваются вычислительные системы жёсткого реального времени (ВС ЖРВ), состоящие из нескольких подсистем, объединённых одной общей средой передачи данных. Особенностью ВС ЖРВ является недопустимость нарушения директивных сроков выполнения задач [1].

В настоящее время методы планирования задач для систем реального времени принято разделять на два класса: статические и динамические [2]. В данной работе вы рассматриваем только статическое планирование. Применение статического планирования возможно лишь в системах, относительно которых уже на стадии их разработки известен состав задач, подлежащих выполнению, и их временные характеристики. Достоинством статического планирования является простота реализации системы и гарантированное выполнение режима реального времени при отсутствии ошибок [1].

Развитие информационных технологий требует от используемых сред передачи данных, объединяющих подсистемы ВС ЖРВ, всё большей пропускной способности. В данной работе в качестве среды передачи данных рассматривается кольцо с арбитражем. Кольцо с арбитражем – это высокоскоростная среда передачи данных, в которой информация передаётся по однонаправленному замкнутому контуру, а возможность передачи данных предоставляется абонентам кольца лишь после выполнения процедуры, называемой ***арбитражем***. Данная процедура гарантирует, что в один и тот же момент времени данные могут передаваться только одним абонентом кольца с арбитражем. Наибольшее распространение в настоящее время получило кольцо с арбитражем стандарта Fibre Channel [3]. Оборудование, работающее по стандарту Fibre Channel, в настоящее время позволяет обеспечивать пропусную способность до 1600 МБ/с.

Особенность кольца с арбитражем заключается в том, что при статическом планировании информационных обменов в условиях высокой загрузки канала необходимо учитывать конечность скорости распространения сигнала в физической среде. Этот фактор учитывается в модели информационных обменов, предложенной в работе [4]. Учёт конечной скорости распространения сигнала в физической среде позволяет более точно учитывать происходящие в кольце с арбитражем процессы и за счёт этого строить расписания с бóльшей загрузкой канала, однако при этом существенное значение приобретает ***порядок абонентов в кольце с арбитражем***, т. е. порядок, в котором абоненты объединены каналами связи в кольцо с арбитражем.

Кольцо с арбитражем не является отказоустойчивой средой передачи данных, поэтому для повышения его надёжности применяют ***концентраторы***. В кольце с арбитражем с концентратором передача данных на логическом уровне происходит по однонаправленному замкнутому контуру, вершинами которого являются абоненты кольца с арбитражем. На физическом уровне передача данных осуществляется по топологии «звезда», в центре которой находится концентратор, к которому подключены абоненты. В случае возникновения неисправности в каком-либо канале связи или в сетевом адаптере, с помощью которого абонент подключен к кольцу с арбитражем, концентратор замыкает неисправный участок сети и тем самым исключает его из логического контура кольца. При этом порядок абонентов в кольце с арбитражем определяется конфигурационными настройками концентратора и порядком подключения кабелей к его гнёздам, поэтому на этапе проектирования системы изменение порядка абонентов в кольце с арбитражем с концентратором не должно вызывать трудностей.

2. Формальная постановка задачи

2.1. Формальная постановка задачи построения расписаний для произвольно заданной функции минимально допустимых задержек

Задачу построения расписаний для произвольно заданной функции минимально допустимых задержек сокращенно будем обозначать символом $\mathbf{T^{SCH}}$. Обозначим $\mathbb{R}^+ = \{\, x \in \mathbb{R} \mid x \geqslant 0 \,\}$.

Определение 1. ***Множеством исходных работ*** *называется непустое конечное множество E^S, на котором определены следующие четыре функции*:

- $s(e)\colon E^S \to \mathbb{R}^+$ – *директивное время начала выполнения работы e;*
- $f(e)\colon E^S \to \mathbb{R}^+$ – *директивное время окончания выполнения работы e;*
- $t(e)\colon E^S \to \mathbb{R}^+$ – *длительность выполнения работы e;*

- $\Delta_{min}(e_a, e_b)\colon E^S \times E^S \to \mathbb{R}^+$ – *функция минимально допустимых задержек.*

Определение 2. ***Расписанием (соответствующим множеству исходных работ*** E^S***)*** *называется множество* $E \subseteq E^S$*, на котором определена функция*

$$s^*(e)\colon E \to \mathbb{R}^+$$

– время начала выполнения работы e*. Причём для множества* E *и для функции* $s^*(e)$ *должны выполняться следующие два условия:*

1) $\forall e \in E \Rightarrow \big(s(e) \leqslant s^*(e)\big) \wedge \big(f^*(e) \leqslant f(e)\big)$ *– интервал выполнения каждой работы должен лежать внутри её директивного интервала;*

2) $\forall e_a \in E,\ \forall e_b \in next(e_a) \Rightarrow s^*(e_a) + \Delta_{min}(e_a, e_b) \leqslant s^*(e_b)$ *– должны выполняться ограничения функции минимально допустимых задержек.*

Здесь использованы следующие обозначения:

- $f^*(e) = s^*(e) + t(e)$ *– время окончания выполнения работы* e*;*
- $next(e)$ *– работа, следующая за работой* e*. Формальное определение:*

$$next(e) = \big\{\, e_b \in (E^S \backslash e_a) \mid \big(s^*(e_b) \geqslant s^*(e_a)\big) \wedge \big(\neg \exists e_x \in (E^S \backslash e_a)\colon s^*(e_a) \leqslant s^*(e_x) < s^*(e_b)\big)\big\}.$$

Задача 1. ***Постановка задачи построения расписаний для произвольно заданной функции минимально допустимых задержек.*** *Задано множество исходных работ* E^S *с определёнными на нём функциями* $t(e)$, $s(e)$, $f(e)$, $\Delta_{min}(e_a, e_b)$. *Требуется найти расписание* $E \subseteq E^S$*, такое что* $|E| \to \max$.

Постановка данной задачи проиллюстрирована на рисунках 1 и 2.

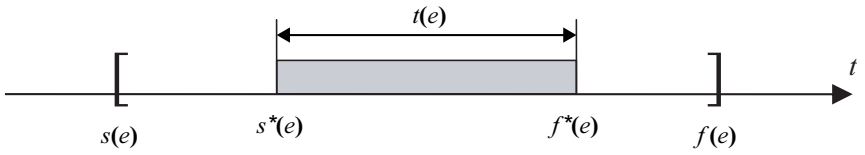

Рис. 1: Основные величины, характеризующие работу.

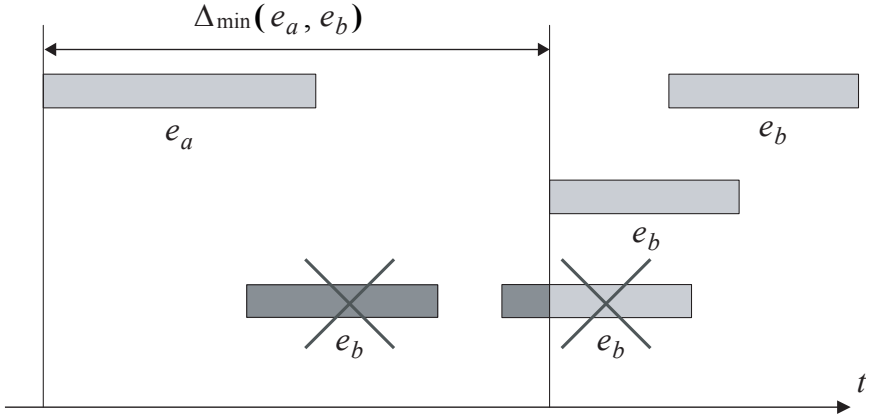

Рис. 2: Минимально допустимая задержка между работами.

2.2. Формальная постановка задачи выбора порядка абонентов в кольце с арбитражем

Задачу выбора порядка абонентов в кольце с арбитражем сокращенно будем обозначать символом $\mathbf{T^{ORD}}$.

Определение 3. Пусть имеется кольцо с арбитражем, в котором число портов равно P ($P \in \mathbb{N}$). Для каждого $i \in \overline{1,P}$ порт с номером i имеет адрес, равный a_i ($a_i \in \mathbb{Z}$). Тогда ***множеством адресов портов*** *данного кольца с арбитражем называется множество*

$$A^{SET} = \{\, a_i \mid a_i \in \overline{1,P} \,\}.$$

Определение 4. Пусть имеется кольцо с арбитражем, в котором множество адресов портов равно A^{SET}. Тогда ***множеством всех возможных порядков абонентов*** *в данном кольце с арбитражем называется множество $R(A^{SET})$ – множество, состоящее из всех упорядоченных перестановок множества A^{SET}.*

Определение 5. Пусть имеется кольцо с арбитражем, в котором множество адресов портов равно A^{SET}. Тогда ***порядком абонентов*** *в данном кольце с арбитражем называется элемент множества $R(A^{SET})$.*

Из данного определения следует, что порядок абонентов – это упорядоченное множество с числом элементов, равным $|A^{SET}|$. Порядок абонентов в кольце с арбитражем имеет следующее смысловое значение: в кольце с арбитражем с порядком абонентом r адрес порта с номером i, где $i \in \overline{1,|A^{SET}|}$, равен $r[i]$, где $[\cdot]$ – операция взятия i-го элемента упорядоченного множества (нумерация начинается с единицы).

Кольцо с арбитражем имеет различные ***параметры***, такие как: P – число портов в кольце с арбитражем, A^{SET} – множество адресов портов в кольце с арбитражем, r – порядок абонентов в кольце с арбитражем, частота передачи данных в кольце с арбитражем, скорость распространения сигнала в физической среде, длины каналов связи от каждого порта до концентратора и др. Совокупность всех этих параметров будем обозначать как $\overline{w} = \{w_i\}_{i=1}^{n_w}$.

Единственный ***изменяемый параметр*** – порядок абонентов в кольце с арбитражем – будем обозначать символом r. Упорядоченное множество ***неизменяемых параметров*** будем обозначать символом $\overline{q}$. Для определённости будем считать, что неизменяемыми являются первые $n_q = (n_w - 1)$ параметров: $\overline{q} = \{q_j\}_{j=1}^{n_q} = \{w_i\}_{i=1}^{(n_w-1)}$. Соответственно, изменяемым является только последний параметр: $r = w_{n_w}$.

Множество всех возможных порядков абонентов $R(A^{SET})$ в векторной записи будем записывать как $R(\overline{q})$.

Определение 6. Пусть M_1 – некоторое множество; $M_2(m_1)$, $M_3(m_1, m_2)$, … , $M_{n-1}(m_1, \ldots, m_{n-2})$, $M_n(m_1, \ldots, m_{n-1})$ – некоторые множества, зависящие от параметров. Тогда ***обобщённым декартовым произведением*** *множеств M_1, … , $M_n(m_1, \ldots, m_{n-1})$ будем называть множество*

$$M_1 \times M_2(m_1) \times M_3(m_1, m_2) \times \ldots \times M_n(m_1, \ldots, m_{n-1}) \stackrel{\text{def}}{=}$$
$$\stackrel{\text{def}}{=} \Big\{ (m_1, \ldots, m_n) \mid m_1 \in M_1,\ m_2 \in M_2(m_1),\ m_3 \in M_3(m_1, m_2), \ldots$$
$$\ldots,\ m_n \in M_n(m_1, \ldots, m_{n-1}) \Big\}.$$

Исходные данные в задаче $\mathbf{T}^{\mathbf{ORD}}$ выбора порядка абонентов в кольце с арбитражем:

- $\overline{q}$ – фиксированное упорядоченное множество значений неизменяемых параметров кольца с арбитражем;
- E^S – непустое конечное множество исходных сообщений. На множестве E^S определены функции:
  - $s(e)\colon E^S \to \mathbb{R}^+$ – директивное время начала выполнения информационного обмена e;
  - $f(e)\colon E^S \to \mathbb{R}^+$ – директивное время окончания выполнения информационного обмена e;
  - $t(\overline{q}, r, e)\colon Q \times R(\overline{q}) \times E^S \to \mathbb{R}^+$ – длительность выполнения информационного обмена e;
  - $\Delta_{min}(\overline{q}, r, e_a, e_b)\colon Q \times R(\overline{q}) \times E^S \times E^S \to \mathbb{R}^+$ – функция минимально допустимых задержек.

Определение 7. ***Индивидуальной задачей $\mathbf{T}^{\mathbf{ORD}}(r)$, получающейся при фиксированном*** *r, будем называть задачу $\mathbf{T}^{\mathbf{SCH}}$, для которой* [1]*:*

- $\widetilde{E^S} = E^S$. *На множестве $\widetilde{E^S}$ определены функции:*
  - $\widetilde{s}(\widetilde{e}) = s(e)$ [2]*;*
  - $\widetilde{f}(\widetilde{e}) = f(e)$*;*
  - $\widetilde{t}(\widetilde{e}) = t(\overline{q}, r, e)$ [3]*;*
  - $\widetilde{\Delta}_{min}(\widetilde{e}_a, \widetilde{e}_b) = \Delta_{min}(\overline{q}, r, e_a, e_b)$.

Формальная постановка задачи $\mathbf{T}^{\mathbf{ORD}}$.

$$\begin{cases} (r, E) = \arg\max(|E|) \\ r \in R(\overline{q}) \\ E \in \mathbf{T}^{\mathbf{ORD}}(r) \end{cases}$$

Здесь запись $E \in \mathbf{T}^{\mathbf{ORD}}(r)$ означает, что расписание информационных обменов E является оптимальным решением задачи $\mathbf{T}^{\mathbf{ORD}}(r)$.

Функции $t(\overline{q}, r, e)$ и $\Delta_{min}(\overline{q}, r, e_a, e_b)$ определяются моделью кольца с арбитражем с концентратором. Эти величины зависят от неизменяемых параметров кольца с арбитражем $\overline{q}$, порядка абонентов в кольце с арбитражем r и атрибутов информационного обмена e (e_a и e_b для функции Δ_{min}). В частности, в используемой нами модели к таким атрибутам относятся:

- $v(e)$ – ***количество слов данных в информационном обмене***;
- $src(e)$ – ***адрес порта-передатчика информационного обмена***;
- $DST(e)$ – ***множество адресов портов-приёмников информационного обмена***.

Формулы для вычисления функций $t(\overline{q}, r, e)$ и $\Delta_{min}(\overline{q}, r, e_a, e_b)$ являются довольно громоздкими, поэтому мы вынуждены их опустить.

[1] Здесь все величины «с волной» относятся к задаче $\mathbf{T}^{\mathbf{ORD}}(r)$, а «без волны» – к задаче $\mathbf{T}^{\mathbf{ORD}}$.

[2] Здесь элемент $\widetilde{e}$ множества $\widetilde{E^S}$ соответствует элементу e множества E^S.

[3] Значение величины $\overline{q}$ задано из условий задачи $\mathbf{T}^{\mathbf{ORD}}$.

3. Предлагаемый алгоритм решения поставленной задачи

3.1. Задача $\mathbf{T^{SCH}}$ *построения расписаний для произвольно заданной функции минимально допустимых задержек*

Задача $\mathbf{T^{SCH}}$ относится к классу NP-трудных задач. Для решения задачи $\mathbf{T^{SCH}}$ хорошие результаты показал приближенный алгоритм $\mathbf{A^{SCH}}$, основанный на методе ветвей и границ. Подробное описание и исследование алгоритма $\mathbf{A^{SCH}}$ планируется в декабрьском номере журнала «Вестник Томского государственного университета» за 2012 год. Приведём здесь кратко принцип работы данного алгоритма.

Процесс работы алгоритма $\mathbf{A^{SCH}}$ представляет собой обход дерева, в вершинах которого располагаются расписания, а каждое ребро соответствует добавлению в расписание одной работы. Поиск начинается с корневой вершины, которой соответствует пустое расписание. Для уменьшения времени работы алгоритма из дерева поиска исключаются рёбра, которые соответствуют добавлению в расписание таких работ, которые, будучи добавлены в расписание на данном шаге, заведомо делают невозможным размещение в расписание по меньшей мере одной из ещё не размещённых работы. Вследствие исключения указанных рёбер из дерева поиска, некоторые вершины могут вовсе не иметь исходящих рёбер.

Алгоритм осуществляет обход дерева «в глубину» в соответствии с приоритетами рёбер, при необходимости возвращаясь по дереву поиска назад. Приоритет рёбер, исходящих из каждой вершины, определяется отношением порядка $\prec^{SEL}$. Если алгоритм на какой-то итерации достиг вершины с глубиной n, то возврат по дереву поиска разрешается лишь до вершины с глубиной $(n - MaxRbk)$, где $MaxRbk$ — параметр алгоритма. Данный параметр позволяет уменьшить вычислительную сложность алгоритма. При $MaxRbk = 0$ возвраты по дереву запрещены, при $MaxRbk = (|E^S| - 1)$ алгоритм может просмотреть всё дерево поиска.

Если на некоторой итерации все ветви дерева поиска, исходящие из данной вершины, оказались тупиковыми, и необходимо вернуться в дереве поиска на шаг назад, однако это невозможно (вследствие того, что текущая вершина является корневой, или в силу ограничения на количество возвратов), то в этом случае алгоритм в соответствии с отношением порядка $\prec^{DEL}$ выбирает одну работу из оставшихся ещё не размещённых работ и удаляет её, после чего все исходящие из текущей вершины рёбра помечаются как непросмотренные, и поиск продолжается далее с текущей вершины.

3.2. Задача $\mathbf{T^{ORD}}$ *выбора порядка абонентов в кольце с арбитражем*

Поскольку для задачи $\mathbf{T^{SCH}}$ известен хороший с практической точки зрения алгоритм решения, то для решения задачи $\mathbf{T^{ORD}}$ имеет смысл использовать некоторый алгоритм поиска по множеству $R(A^{SET})$ всех возможных порядков абонентов, в котором значение целевой функции для соответствующего порядка абонентов r будет вычисляться алгоритмом $\mathbf{A^{SCH}}$.

Для решения задач оптимизации, в которых целевая функция задана алгоритмически и её структура априорно неизвестна, хорошо зарекомендовали себя генетические алгоритмы [5], поэтому для решения задачи $\mathbf{T^{ORD}}$ мы использовали генетический (эволюционный) алгоритм.

Генетические алгоритмы являются некоторым шаблоном, требующим индивидуального наполнения для каждой конкретной задачи. Приведём далее используемые нами

Таблица 1. Числовые параметры алгоритма $\mathbf{A}^{\mathbf{ORD}}$.

Параметр	Допустимые значения	Описание
max_iters	$\mathbb{N}$	Максимальное число итераций алгоритма.
pop_size	$\mathbb{N} \setminus \{1\}$	Размер популяции.
$mating\_pool\_part$	$[2/pop\_size, 1]$	Размер родительского пула (в долях от размера популяции).
$tour\_size$	$\mathbb{N}$	Размер турнира.
$tour\_sel\_prob$	$[0, 1]$	Вероятность выбора очередной особи в турнире.
$cross\_prob$	$[0, 1]$	Вероятность выполнения операции скрещивания.
mut_prob	$[0, 1]$	Вероятность выполнения операции мутации.
$elite\_select\_part$	$[0, 1]$	Доля элитных особей при селекции от размера родительского пула.
$elite\_remain\_part$	$[0, 1]$	Доля элитных особей при формировании новой популяции от размера популяции.

параметры этого шаблона.

Схема генетического алгоритма. В алгоритме $\mathbf{A}^{\mathbf{ORD}}$ мы использовали классическую схему [6]:

1) Сгенерировать случайным образом популяцию размера pop_size.
2) Вычислить целевую функцию для каждой особи популяции, для которой целевая функция ещё не вычислена.
3) Если критерий останова достигнут, завершить работу алгоритма.
4) Выполнить селекцию особей.
5) Выполнить скрещивание особей.
6) Выполнить мутации особей.
7) Сформировать новую популяцию.
8) Перейти к шагу 2.

Числовые параметры алгоритма перечислены в таблице 1. Смысловое значение данных параметров поясняется далее. Для дальнейшего изложения введём следующие обозначения:

- $mating\_pool\_size = mating\_pool\_part \cdot pop\_size$ – количество особей в родительском пуле;
- $elite\_select\_count = elite\_select\_part \cdot mating\_pool\_size$ – количество элитных особей при селекции;
- $elite\_remain\_count = elite\_remain\_part \cdot pop\_size$ – количество элитных особей при формировании новой популяции.

Способ кодирования особей. Поскольку пространство поиска представляет собой множество всех перестановок, состоящих из элементов множества A^{SET}, то имеет смысл кодировать особи в виде упорядоченного множества длины $|A^{SET}|$. Данный способ кодирования эквивалентен кодированию особей в виде строки фиксированной длины $|A^{SET}|$ из алфавита A^{SET}.

Целевая функция вычисляется алгоритмом $\mathbf{A^{SCH}}$.

Критерии останова.

- нахождение порядка абонентов r, для которого алгоритм $\mathbf{A^{SCH}}$ построил полное расписание;
- выполнение max_iters итераций.

Операция создания случайной особи. Особь (упорядоченное множество) r формируется из всех элементов множества A^{SET}, расположенных в случайном порядке, без повторов.

Селекция особей. В результате селекции из всей популяции выбираются особи, которые образуют ***родительский пул***. Селекция особей производится путём многократного выполнения операции селекции.

В качестве операции селекции была выбрана ***турнирная селекция***, поскольку в целом она даёт более хорошие результаты, чем пропорциональная селекция, и не уступает селекции линейным ранжированием, имея при этом меньшую вычислительную сложность, чем селекция линейным ранжированием [7]. Селекция Genitor не была нами выбрана, поскольку она даёт хорошие результаты преимущественно на популяциях больших размеров [там же].

Приведём описание операции турнирной селекции. Из популяции случайным образом выбираются $tour\_size$ особей. Эти особи упорядочиваются по убыванию целевой функции. С вероятностью $tour\_sel\_prob$ выбирается лучшая особь. Если лучшая особь не выбрана, то с вероятностью $tour\_sel\_prob$ выбирается следующая за ней лучшая особь и т. д. Выбранная таким образом особь добавляется в родительский пул.

Мы использовали элитную стратегию, согласно которой в родительский пул обязательно выбираются $elite\_select\_count$ лучших особей, а остальные $(mating\_pool\_size - elite\_select\_count)$ особей выбираются с помощью турнирной селекции. Элитные особи и особи, добавленные в родительский пул в результате турнирной селекции, из последующих турниров на данной итерации алгоритма исключаются.

Скрещивание особей. Скрещивание особей выполняется следующим образом. Случайным образом из родительского пула выбирается пара особей. Затем с вероятностью $cross\_prob$ выполняется пункт (а), в противном случае выполняется пункт (б):

а) К двум выбранным особям применяется операция скрещивания, в результате чего получается пара особей-потомков. Особи-потомки добавляются в пул потомков.

б) Выбранные особи добавляются в пул потомков.

Данная операция повторяется до тех пор, пока пул потомков не достигнет размера $(pop\_size - elite\_remain\_count)$.

В данной работе рассматриваются следующие три широко известные операции скрещивания перестановок: Order Crossover (OX) [8], Cycle Crossover (CX) [9] и Partially Mapped Crossover (PMX) [10].

Мутация особей. Мутация особей выполняется следующим образом. Для каждой особи из пула потомков с вероятностью mut_prob выполняется операция мутации. Особь, получившаяся в результате операции мутации, замещает в популяции исходную особь.

В данной работе рассматривались следующие две операции мутации: Reverse Sequence Mutation (RSM) [11] и Crossover with Random String Mutation (CRM). Операция CRM заключается в скрещивании заданной особи со случайно созданной особью. Во всех проведённых в данной работе экспериментах операция скрещивания в CRM использовалась та же самая, что и в самом генетическом алгоритме.

Формирование новой популяции. Новую популяцию составляют все ($pop\_size - elite\_remain\_count$) особей из пула потомков, а также $elite\_remain\_count$ лучших особей из исходной популяции.

4. Численное исследование алгоритма $\mathbf{A}^{\mathbf{ORD}}$

4.1. Подбор оптимальных параметров алгоритма методом покоординатного спуска.

Для данного исследования было сгенерировано множество исходных работ E_1^S со следующими интегральными характеристиками: число исходных работ $|E_1^S| = 100$, коэффициент загрузки канала $K(E_1^S) = 0{,}425$, средняя степень свободы работ $D(E_1^S) = 4$, где

$$K(E^S) = \frac{\sum\limits_{e \in E^S} t(e)}{\max\limits_{e \in E^S} f(e) - \min\limits_{e \in E^S} s(e)}, \quad D(E^S) = \frac{1}{|E^S|} \cdot \sum_{e \in E^S} \frac{f(e) - s(e) - t(e)}{t(e)}.$$

При таких характеристиках исходные данные не являются слишком «простыми», т. е. с вероятностью близкой к единице не позволяют построить полное расписание при помощи перебора небольшого количества вариантов случайно сгенерированных порядков абонентов, а с другой стороны не являются слишком «сложными», что позволило нам выполнить достаточное большое число запусков генетического алгоритма за приемлемое с практической точки зрения время. Для множества исходных работ E_1^S алгоритм $\mathbf{A}^{\mathbf{SCH}}$ с параметром $MaxRbk = 10$ строил полное расписание, поэтому в генетическом алгоритме при вычислении целевой функции алгоритмом $\mathbf{A}^{\mathbf{SCH}}$ использовалось значение параметра $MaxRbk = 10$.

В качестве начальных значений параметров алгоритма были выбраны значения, предположительно показывающие хорошие результаты (часть значений параметров была взята из работы [12]). Начальные значения параметров приведены в таблице 2. Параметры $mating\_pool\_size$ и $elite\_remain\_count$ вычислялись как доля от размера популяции pop_size, параметр $elite\_select\_count$ – как доля от размера родительского пула $mating\_pool\_size$. Параметр max_iters во всех случаях был равен $\lfloor 500/pop\_size \rfloor$, чтобы количество вычислений целевой функции во всех случаях не превышало 500.

Метод покоординатного спуска мы использовали следующим образом. Из всей совокупности оптимизируемых параметров выбирался один параметр. Данный выбранный параметр пробегал по множеству исследуемых значений (см. таблицу 2). Для каждого значения выбранного параметра алгоритм $\mathbf{A}^{\mathbf{ORD}}$ запускался 5 раз, и вычислялись

Таблица 2. Начальные и конечные значения параметров при исследовании алгоритма $\mathbf{A}^{\mathbf{ORD}}$ методом покоординатного спуска.

Параметр	Начальное значение	Исследуемые значения	Конечное значение
pop_size	25	5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100	50
mating_pool_part	0,8	от 0,1 до 1 с шагом 0,1	0,1
tour_size	2	1, 2, 4, 8, 16	8
tour_sel_prob	0,8	от 0 до 1 с шагом 0,1	1,0
cross_prob	0,4	от 0 до 1 с шагом 0,1	0,6
mut_prob	0,1	от 0 до 1 с шагом 0,1	0,9
elite_select_part	0,2	от 0 до 1 с шагом 0,1	0,2
elite_remain_part	0,1	от 0 до 1 с шагом 0,1	0,6
crossover	OX	OX, CX, PMX	OX
mutation	RSM	RSM, CRM	RSM

средние значения целевой функции и количества подсчётов целевой функции алгоритмом $\mathbf{A}^{\mathbf{SCH}}$. Лучшим значением выбранного параметра считалось то, при котором значение целевой функции было максимальным. Причём при равенстве значений целевых функций лучшим значением выбранного параметра считалось то, при котором количество запусков алгоритма $\mathbf{A}^{\mathbf{SCH}}$ было минимальным.

Значение выбранного параметра присваивалось наилучшему найденному значению, после чего выбирался другой параметр. Параметры выбирались циклически, в порядке, указанном в таблице. Вычисления останавливались, когда ни один из параметров более не изменялся. Результаты исследования приведены в таблице 2.

4.2. Сравнение скорости работы генетического алгоритма с алгоритмом случайного поиска

Был реализован алгоритм случайного поиска, который на каждой итерации генерировал случайный порядок абонентов и с помощью алгоритма $\mathbf{A}^{\mathbf{SCH}}$ вычислял для него значение целевой функции. Было произведено 100 запусков генетического алгоритма с параметрами, указанными в таблице 2, и 100 запусков алгоритма случайного поиска. Для каждого из алгоритмов вычислялось количество запусков алгоритма $\mathbf{A}^{\mathbf{SCH}}$ до построения полного расписания. Результаты исследования следующие: 238 запусков для генетического алгоритма и 401 запуск для алгоритма случайного поиска. Таким образом, на исследованных исходных данных генетический алгоритм находил порядок абонентов с построением полного расписания в среднем в 1,7 раз быстрее, чем алгоритм случайного поиска.

5. Заключение

В данной работе описан разработанный генетический алгоритм решения задачи выбора порядка абонентов в кольце с арбитражем. В работе описаны: способ кодирования особей, способ вычисления целевой функции, числовые параметры алгоритма,

подходящие операции скрещивания и мутации. Методом покоординатного спуска получены значения параметров алгоритма, на которых алгоритм показывает хорошую скорость сходимости к оптимуму. Проведенное сравнение разработанного генетического алгоритма с алгоритмом случайного перебора показало, что среднее число запусков алгоритма $\mathbf{A}^{\mathbf{SCH}}$, необходимых для построения полного расписания, для генетического алгоритма меньше чем для алгоритма случайного поиска в 1,7 раза.

Дальнейшие исследования предложенного генетического алгоритма могут включать нахождение значений его оптимальных параметров методами, сочетающими локальный и глобальный поиск. Данное исследование, предположительно, потребует наличия значительных вычислительных мощностей.

СПИСОК ЛИТЕРАТУРЫ

1. *Данилов М. В.* Методы планирования выполнения задач в системах реального времени // Программные продукты и системы. 2001. №4. С. 28–35.

2. *Stankovic J. A. et al.* Implications of Classical Scheduling Results For Real-Time Systems // IEEE Computer. 1995. Vol. 28, No. 6. P. 16–25.

3. ISO/IEC 14165-122:2005. Information technology – Fibre Channel – Part 122: Arbitrated Loop-2 (FC-AL-2) // International Organization for Standardization (ISO), 2005.

4. *Бычков И. А.* Представление математических моделей сред передачи данных жёсткого реального времени для задач статического планирования // Математическое моделирование. 2011. Т. 23. №12. С. 117–131.

5. *Костенко В. А.* Принципы построения генетических алгоритмов и их использование для решения задач оптимизации // Труды IV Международной конференции «Дискретные модели в теории управляющих систем» (19-25 июня 2000 г.). С. 49-55.

6. *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И. Д. Рудинского. М.: Горячая линия-Телеком, 2006.

7. *Goldberg D. E., Deb K.* A comparative analysis of selection schemes used in genetic algorithms // Foundations of Genetic Algorithms. 1991. P. 69–93.

8. *Davis L.* Job-shop Scheduling with Genetic Algorithms // Proceedings of an International Conference on Genetic Algorithms and Their Applications. 1985. P. 136–140.

9. *Oliver I. M., Smith D. J., Holland, J. R. C.* A study of permutation crossover operators on the traveling salesman problem // In Proceedings of the second international conference. on genetic algorithms (ICGA'87). 1987. P. 224–230.

10. *Goldberg D. E., Lingle R.* Alleles, loci, and the traveling salesman problem // In Proceedings of the International Conference on Genetic Algorithms and Their Applications. 1985. P. 154–159.

11. *Abdoun O., Abouchabaka J., Tajani C.* Analyzing the Performance of Mutation Operators to Solve the Travelling Salesman Problem // International Journal of Emerging Sciences. 2012. Vol. 2. No. 1. P. 61–77.

12. *Костенко В. А., Смелянский Р. Л., Трекин А. Г.* Синтез структур вычислительных систем реального времени с использованием генетических алгоритмов // Программирование. 2000. №5. С. 63–72.