DOI: 10.25728/mlsd.2023.1138

# РЕГИОНАЛЬНОЕ УПРАВЛЕНИЕ: ОПЕРАТИВНОЕ ПЛАНИРОВАНИЕ В РЕЖИМЕ РЕАЛЬНОГО ВРЕМЕНИ

**Кононов Д.А.**
*Российский государственный гуманитарный университет*
*Институт проблем управления им. В.А. Трапезникова РАН, Москва, Россия*
dmitrykon52@gmail.com
**Фуругян М.Г.**
*Федеральный исследовательский центр «Информатика и управление»*
*Российской академии наук, Москва, Россия*
rtsccas@yandex.ru

*Аннотация. Рассматривается задача планирования комплексов работ, требования на выполнение которых в региональной системе управления поступают в заданные моменты времени. Характеристики работ (длительности и директивные сроки) становится известными в момент поступления каждого запроса. Для выполнения работ используются исполнительные механизмы. Разработан метод оперативного построения допустимого расписания.*

*Ключевые слова: региональное управление, планирование работ, исполнительный механизм, допустимое расписание, сетевая модель, максимальный поток.*

## Введение

Проблемы распределения ресурсов и календарного планирования возникают в различных областях деятельности человека. Так, в региональных системах управления, указанные задачи широко распространены при проектировании, испытаниях и эксплуатации сложных технических объектов (самолеты, ракеты, электростанции), при проведении опытно-конструкторских работ, в гражданском и военном строительстве, при оценке запасов полезных ископаемых в месторождениях, при обработке больших массивов информации, при проектировании и функционировании транспортных и конвейерных систем, во многих других областях. В системах реального времени каждая работа должна выполняться в строго заданном временном директивном интервале и завершиться не позднее установленного заранее срока. При этом одна из основных задач заключается в распределении ресурсов между работами и построении оптимального расписания их выполнения. Алгоритмам решения таких задач посвящено большое количество публикаций. Здесь можно отметить такие фундаментальные работы, как [1–3], в которых авторы исследуют различные постановки (составление расписаний с прерываниями и переключениями с одного исполнителя на другой и без прерываний, задачи на быстродействие и на соблюдение директивных сроков, построение однопроцессорных и многопроцессорных расписаний). В [3] исследуются NP-трудные задачи быстродействия и минимизации максимального временного смещения для одного и нескольких приборов. Предлагается новый подход к поиску приближенных решений. В [4, 5] предлагается методика построения оптимальных расписаний в задачах с нефиксированными параметрами (длительности, потребляемые ресурсы). Методика основана на использовании схемы “ветвей и границ” и построении многогранников устойчивости решений. В [6–8] разработана методика проверки выполнения ограничений реального времени, заключающихся в том, что каждая работа должна выполняться в заданном директивном интервале. Проведенные исследования основаны на построении имитационной модели с использованием обобщенных конечных автоматов с остановкой таймера. С помощью этой модели строится временная диаграмма работы системы, позволяющая осуществить непосредственную проверку выполнения ограничений реального времени. В [9] предложен псевдополиномиальный алгоритм решения задачи построения оптимального по быстродействию расписания исполнения заданий с логическими условиями предшествования. В этой задаче для каждого задания дан список его непосредственных предшественников, а также число завершенных непосредственных предшественников, необходимое для начала его выполнения. Задача сведена к циклической игре. В [10, 11] некоторые задачи планирования работ сведены к минимаксным задачам.

Указанные выше публикации посвящены распределению возобновляемых ресурсов (процессоров, машин, исполнительных механизмов, приборов, рабочих), т.е. ресурсов, которые могут использоваться многократно. Ряд публикаций посвящен распределению невозобновляемых ресурсов (финансы, топливо, электроэнергия, различные материалы). В отличие от возобновляемых ресурсов, невозобновляемые повторно использоваться не могут. В этой связи отметим работы [12, 13], в которых

предполагается, что длительности заданий линейно зависят от величины выделенного им ресурса. В [14–17] исследована задача со смешанными типами ресурсов – возобновляемыми и невозобновляемыми. Для задачи составления допустимого расписания с директивными интервалами разработаны полиномиальные алгоритмы, основанные на сведéнии исходной задачи к потоковой в сети специального вида.

В настоящей статье рассматривается задача планирования комплексов работ, требования на выполнение которых поступают в фиксированные моменты времени. Для выполнения работ используются исполнительные механизмы. Работы допускают прерывания и переключения с одного исполнителя на другой. Характеристики работ (длительности и директивные сроки) становится известными в момент поступления каждого запроса. Поэтому возникает необходимость составлять расписание выполнения комплексов работ оперативно, т.е. в режиме реального времени. Это, в свою очередь, требует разработки эффективных алгоритмов планирования. Разработан полиномиальный алгоритм оперативного построения допустимого расписания.

Рассмотрена также задача с нефиксированными длительностями выполнения работ, которые могут принимать целые значения из заданных временных интервалов. Совокупность таких интервалов образует многомерный параллелепипед, каждая точка которого определяет некоторый вектор длительностей выполнения работ, который становится известным только в ходе проведения эксперимента, т.е. в реальном масштабе времени. Поэтому предлагается проводить расчет расписаний заранее, т.е. до проведения эксперимента, для всевозможных значений длительностей заданий. Однако, при большом числе работ сделать это практически невозможно. Задача заключается в том, чтобы разбить параллелепипед на области, внутри каждой из которых расписание не изменяет своей структуры. В то же время, число таких областей должно быть существенно меньше числа целочисленных точек параллелепипеда.

## 1. Математическая постановка задачи

В моменты времени $\tau_k$ поступают требования на выполнение комплексов работ (заданий) $W_k = \{w_k^1, w_k^2, \dots, w_k^{r_k}\}$, $k = \overline{1,K}$, $\tau_1 < \tau_2 < \dots < \tau_K$. Для выполнения работ в каждом интервале $[\tau_k, \tau_{k+1}]$ имеется $m_k$ исполнительных механизмов (или, просто, исполнителей), $k = \overline{1,K-1}$, Все исполнители идентичные. Каждое задание $w_k^i$ имеет следующие характеристики: $t_k^i$ – длительность, $[b_k^i, c_k^i]$ – директивный интервал (работа $w_k^i$ может выполняться только в этом интервале), $b_k^i \geq \tau_k$, $k = \overline{1,K}$, $i = \overline{1,r_k}$. При выполнении заданий допускаются их прерывания и переключения с одного исполнителя на другой, которые по предположению не требуют временных затрат. Кроме того, не допускается параллельное выполнение одного задания несколькими исполнителями и одновременное выполнение нескольких работ одним исполнителем.

Моменты $\tau_k$, $k = \overline{1,K}$, известны заранее. Однако, состав комплекса заданий $W_k$ и их характеристики становятся известными только в момент поступления запроса на его выполнение, т.е. в момент $\tau_k$,$k = \overline{1,K}$. В этом случае планировать выполнение работ $W_k$ возможно только после момента $\tau_k$, $k = \overline{1,K}$, т.е. в режиме реального времени.

Требуется определить, существует ли допустимое расписание для всего комплекса работ $W = \bigcup_{k=1}^{K} W_k$ (т.е. расписание, при котором каждое задание выполняется в своем директивном интервале) и построить его в случае положительного ответа. Предполагается, что временем работы самого алгоритма построения расписания можно пренебречь.

Решение поставленной задачи основано на построении сетевой потоковой модели и поиске максимального потока.

## 2. Построение допустимого расписания

### 2.1. Построение сетевой модели и расписания для $W_1$

Пусть $y_1^0 < y_1^1 < \dots < y_1^{p_1}$ – все различные величины среди $\tau_1$, $\tau_2$, $b_1^i$, $c_1^i$, $i = \overline{1,r_1}$, принадлежащие интервалу $[\tau_1, \tau_2]$. Определим интервалы $I_1^j = \left[y_1^{j-1}, y_1^j\right]$, $j = \overline{1,p_1}$, и сеть $G_1 = (V_1, A_1)$, где $V_1 = \left\{u, v, I_1^j, j = \overline{1,p_1}, w_1^i, i = \overline{1,r_1}\right\}$ – множество узлов, $A_1 = \{\left(u, I_1^j\right)$, $j = \overline{1,p_1}$, $\left(I_1^j, w_1^i\right), j = \overline{1,p_1}, i = \overline{1,r_1}$, $\left(w_1^i, v\right), i = \overline{1,r_1}\}$ – множество ориентированных дуг. Дуга $\left(I_1^j, w_1^i\right)$ вводится в сеть, если $I_1^j \subseteq [b_1^i, c_1^i]$. Отметим, что при всех $j$ и $i$ либо $I_1^j \subseteq [b_1^i, c_1^i]$, либо

$I_1^j \cap [b_1^i, c_1^i] = \emptyset$. Пропускные способности $U_1$ дуг определяются следующим оборазом: $U_1(u, I_1^j) = m\delta_1^j$, $U_1(I_1^j, w_1^i) = \delta_1^j$, $U_1(w_1^i, v) = t_1^i, j = \overline{1, p_1}, i = \overline{1, r_1}$; $\delta_1^j = y_1^j - y_1^{j-1}$.

При нахождении максимального потока $f_1$ в сети $G_1$ используется следующая модификация алгоритма А.В. Карзанова [18]. При проталкивании потока из вершины $I_1^j$ вправо в первую очередь следует использовать дуги $(I_1^j, w_1^i)$ для работ $w_1^i$, у которых $c_1^i \le \tau_2$. Это объясняется тем, что такие задания могут выполняться только в интервале $[\tau_1, \tau_2]$, а работам с директивным сроком, большим $\tau_2$, процессорное время может выделяться и после момента $\tau_2$.

Величина потока $f_1(I_1^j, w_1^i)$ равна объему процессорного времени, выделенному работе $w_1^i$ в интервале $I_1^j$. Расписание в интервале $I_1^j$ строится с помощью алгоритма упаковки.

Если хотя бы для одной работы $w_1^i$ с директивным сроком $c_1^i \le \tau_2$ оказалось, что $f_1(w_1^i, v) < t_1^i$ (т.е. дуга $(w_1^i, v)$ насыщена не полностью), то допустимого расписания для $W_1$, и, следовательно, для всего комплекса заданий $W$, не существует. В случае, когда $f_1(w_1^i, v) = t_1^i$ для всех работ $w_1^i$ с директивным сроком $c_1^i \le \tau_2$, построение допустимого расписания продолжится в момент $\tau_2$ для незавершенных работ из $W_1$ (если таковые имеются) и вновь поступивших работ из $W_2$.

### 2.1. Построение сетевой модели и расписания для $W_1 \cup \ldots \cup W_k, k = \overline{2, K}$.

Предположим, что построена потоковая сеть $G_{k-1}, k = \overline{2, K}$, в которой найден максимальный поток $f_{k-1}$. Если хотя бы для одной работы $w_r^i, r = \overline{2, k-1}$, для которой $c_r^i \le \tau_k$, после нахождения максимального потока $f_{k-1}$ в сети $G_{k-1}$ оказалось, что $f_{k-1}(w_r^i, v) < t_r^i$, т.е. дуга $(w_r^i, v)$ насыщена не полностью, то допустимого расписания для $W_1 \cup \ldots \cup W_{k-1}$, и, следовательно, для всего комплекса $W$, не существует. Если же $f_{k-1}(w_r^i, v) = t_r^i$ для всех дуг $(w_r^i, v)$, $r = \overline{1, k-1}$, для которых $c_r^i \le \tau_{k-1}$, то построение допустимого расписания для $W$ продолжается в момент $\tau_k$ для незавершенных работ из множества $W_1 \cup \ldots \cup W_{k-1}$ (если таковые имеются) и вновь поступивших заданий из $W_k$.

Пусть $y_k^0 < y_k^1 < \cdots < y_k^{p_k}$ – все различные величины $\tau_k$, $\tau_{k+1}$, $b_k^i$, $c_k^i$, $i = \overline{1, r_k}$. Определим интервалы $I_k^j = [y_k^{j-1}, y_k^j]$, и пусть $\delta_k^j = y_k^j - y_k^{j-1}$. Из сети $G_{k-1}$ удалим следующие элементы:

- узлы $I_{k-1}^j, j = \overline{1, p_{k-1}}$, и все инцидентные им дуги $(u, I_{k-1}^j)$, $(I_{k-1}^j, w_{k-1}^i)$, $j = \overline{1, p_k}$, $i = \overline{1, r_{k-1}}$; $i = \overline{1, r_{k-1}}$
- узлы $w_z^i$, для которых

$$f_{k-1}(w_z^i, v) = t_z^i \quad (1)$$

и соответствующие дуги $(w_z^i, v)$, $z = \overline{1, k-1}$, $i = \overline{1, r_z}$ (поскольку выполнение условия (1) означает завершение работы $w_z^i$).

Далее, длительности каждой оставшейся работы $w_z^i$ уменьшаются на величину потока $f_{k-1}(w_z^i, v)$ по дуге $(w_z^i, v)$, $z = \overline{1, k-1}$, $i = \overline{1, r_z}$.

Сеть $G_k$ строится из оставшейся не удаленной части сети $G_{k-1}$ путем добавления к ней узлов $I_k^j$, $w_k^i$ и дуг $(I_k^j, w_z^i)$, $j = \overline{1, p_k}$, $i = \overline{1, r_z}$, $z = \overline{1, k}$, $(w_k^i, v)$, $i = \overline{1, r_k}$ по аналогии с тем, как это описано в [1] при $m = m_k$, $n = n_k +$ (*число оставшихся вершин в сети* $G_{k-1}$).

Далее, в сети $G_k$ находится максимальный поток $f_k$ с помощью модификации алгоритма А.В. Карзанова. А именно, при проталкивании потока из вершины $I_k^j$ вправо в первую очередь следует использовать дуги $(I_k^j, w_k^i)$ для работ $w_k^i$, у которых $c_k^j \le \tau_{k+1}$. Величина потока $f_k(I_k^j, w_k^i)$ равна объему процессорного времени, выделяемому работе $w_k^i$ в интервале $I_k^j$. Расписание выполнения работ в интервале $I_k^j$ находится с помощью алгоритма упаковки [1].

Вычислительная сложность предложенного алгоритма составляет $O\left(K\left(\sum_{k=1}^K r_k\right)^3\right)$.

## 3. Планирование работ с неопределенными длительностями

Планируется выполнения комплекса работ (заданий) $W = \{1, 2, \ldots, n\}$, занумерованных от 1 до $n$. Длительность $t_i$ задания $i \in W$ может принимать натуральные значения из интервала

$[t_i^1, t_i^2], t_i^1, t_i^2 \in N, i = \overline{1,n}$, $N$–множество натуральных чисел. Пусть $N_i = \{t_i^1, t_i^1 + 1, \ldots, t_i^2\}$ – возможные значения длительностей задания $i \in W, \Omega_0 = N_1 \times N_2 \times \ldots \times N_n$. Предполагается, что проводится некоторый эксперимент (например, испытание самолета), в ходе которого вычисляются некоторые параметры, определяющие длительности выполнения заданий. Таким образом, характеристики работ становятся известными только в ходе проведения эксперимента, т.е. в режиме реального времени. Основная задача, которую требуется решить – это составить допустимое расписание выполнения заданий $W$ (т.е. такое расписание, при котором каждая работа $i \in W$ выполняется строго в своем директивном интервале) или определить, что такого расписания не существует. Есть несколько путей решения этой задачи. Первый – строить расписание в режиме реального времени, т.е. сразу после того, как станут известными длительности выполнения работ. Однако, это может привести к непозволительным задержкам при проведении эксперимента. Второй подход заключается в предварительном (до проведения эксперимента) построении расписания для каждого целочисленного временного вектор $(\tau_1, \tau_2, \ldots, \tau_n) \in \Omega_0$. Однако, число таких векторов, равное $\prod_{i=1}^{n} T_i$, $(T_i = t_i^2 - t_i^1)$, может быть очень велико, что делает невозможным и этот подход. Мы предлагаем разбить множество $\Omega_0$ на подмножества (число которых существенно меньше числа точек в $\Omega_0$), такие, что в пределах каждого такого подмножества структура допустимого расписания остается неизменной. В этом случае его можно будет вычислять до проведения эксперимента для одного временного вектора из каждого подмножества.

Для фиксированного временного вектора $(\tau_1, \tau_2, \ldots, \tau_n) \in \Omega_0$ допустимое расписание может быть найдено с помощью известных алгоритмов. Например, с помощью псевдополиномиального алгоритма в случае, когда работы не допускают прерываний и переключений с одного процессора на другой [19]. Для случая, когда работы допускают прерывания и переключения, может быть использован полиномиальный алгоритм [1, 20]. Вычислительная сложность указанных алгоритмов составляет $O(nT^m)$и $O(pn^3)$соответственно, где $T = \max_{i=\overline{1,n}} T_i$, $m$ – число процессоров, $p$ – число различных типов процессоров.

### 3.1. Разбиение множества $\Omega_0$ на подмножества

Независимо от того, какая именно задача составления расписания рассматривается, для $\tau \in \Omega_0$ введем обозначения: $f(\tau) = 0$, если допустимого расписания не существует, и $f(\tau) = 1$ в противном случае, и пусть $R(\tau)$–допустимое расписание, соответствующее временному вектору $\tau$. Иными словами, предполагается, что имеется некоторая программа $P(\tau)$, позволяющая для каждого временного вектора $\tau \in \Omega_0$ решать задачу поиска допустимого многопроцессорного расписания с директивными интервалами.

У т в е р ж д е н и е 1. Если $(\tau_1, \tau_2, \ldots, \tau_n) \in \Omega_0$ и $f(\tau_1, \tau_2, \ldots, \tau_n) = 0$, то $f(a_1, a_2, \ldots, a_n) = 0$, где $(a_1, a_2, \ldots, a_n) \in \Omega_0$, $a_j \geq \tau_j, j = \overline{1,n}$.

Д о к а з а т е л ь с т в о. Действительно, если для некоторого временного вектора $\tau = (\tau_1, \tau_2, \ldots, \tau_n) \in \Omega_0$ допустимого расписания не существует, то и для вектора, компоненты которого не меньше компонент вектора $\tau$, его также не существует.

У т в е р ж д е н и е 2. Если $(\tau_1, \tau_2, \ldots, \tau_n) \in \Omega_0$ и $f(\tau_1, \tau_2, \ldots, \tau_n) = 1$, то $f(b_1, b_2, \ldots, b_n) = 1$, где $(b_1, b_2, \ldots, b_n) \in \Omega_0$, $b_j \leq \tau_j, j = \overline{1,n}$.

Д о к а з а т е л ь с т в о. Действительно, если для некоторого временного вектора $\tau = (\tau_1, \tau_2, \ldots, \tau_n) \in \Omega_0$ допустимое расписание существует, то и для вектора, компоненты которого не превосходят компонент вектора $\tau$, оно также существует. Отметим также, что если в расписании для каждого задания указывать процессоры, которые его выполняют, и соответствующие временные интервалы, то для вектора $(b_1, b_2, \ldots, b_n)$ можно использовать то же расписание, которое построено для вектора $(\tau_1, \tau_2, \ldots, \tau_n)$, т.е. расписание с той же структурой. Отличие заключается только в том, что в расписании $R(b_1, b_2, \ldots, b_n)$ некоторые работы могут завершаться раньше, чем в расписании $R(\tau_1, \tau_2, \ldots, \tau_n)$.

Введем обозначения:$\tau_i^0$ – ближайшее целое к значению $\frac{t_i^2 - t_i^1}{2}, i = \overline{1,n}$. Фиксируем произвольное $k, 1 \leq k \leq n$, и рассмотрим функцию $f(\tau_1^0, \ldots, \tau_{k-1}^0, \tau_k, \tau_{k+1}^0, \ldots, \tau_n^0)$ , $\tau_k \epsilon [t_k^1, t_k^2] \cap N$, как функцию одной переменной $\tau_k$. С помощью дихотомической процедуры деления отрезка $[t_k^1, t_k^2]$ пополам находим

$\tau_k^0 \in [t_k^1, t_k^2] \cap N$, такое, что $f(\tau_1^0, \dots, \tau_{k-1}^0, \tau_k^0, \tau_{k+1}^0, \dots, \tau_n^0) = 1$, $f(\tau_1^0, \dots, \tau_{k-1}^0, \tau_k^0 + 1, \tau_{k+1}^0, \dots, \tau_n^0) = 0$. Отметим, что при этом число обращений к программе $P(\tau)$ составляет $O(\log T)$.

В результате, из утверждений 1 и 2 получим, что

$$f(\tau_1, \tau_2, \dots, \tau_n) = 1 \quad (2)$$

при $\tau_j \le \tau_j^0$, $j = \overline{1,n}$, и для временного вектора $(\tau_1, \tau_2, \dots, \tau_n)$ можно использовать расписание, построенное для вектора $(\tau_1^0, \tau_2^0, \dots, \tau_n^0)$. Кроме того,

$$f(\overline{\tau}_1, \overline{\tau}_2, \dots, \overline{\tau}_n) = 0 \quad (3)$$

при $\overline{\tau}_j \ge \tau_j^0, j = \overline{1,n}, j \ne k; \overline{\tau}_k \ge \tau_k^0 + 1$.

Из (2), (3) следует, что множество временных векторов $\Omega_0$ разбилось на четыре подмножества: $\Omega_1^1, \Omega_1^2, \overline{\Omega}_1^1, \overline{\Omega}_1^2$со следующими характеристиками.

1) Для множеств $\Omega_1^1$ и $\Omega_1^2$ вопрос о существовании допустимого расписания остается открытым.

2) $f(\tau) = 1$ при $\tau \in \overline{\Omega}_1^1$, т.е. для $\tau \in \overline{\Omega}_1^1$ допустимое расписание существует и имеет такую же структуру, как и для вектора $(\tau_1^0, \dots, \tau_k^0, \dots, \tau_n^0)$.

3) $f(\tau) = 0$ при $\tau \in \overline{\Omega}_1^2$, т.е. для $\tau \in \overline{\Omega}_1^2$ допустимого расписания не существует.

Таким образом, в результате указанных выше действий для половины временных векторов из $\Omega_0$ (т.е. для $\tau \in \overline{\Omega}_1^1 \cup \overline{\Omega}_1^2$) вопрос о существовании допустимого расписания и его построении в случае положительного ответа будет решен. Вопрос остается открытым для двух множеств – $\Omega_1^1$ *и* $\Omega_1^2$, суммарное количество целочисленных векторов в которых вдвое меньше числа векторов в $\Omega_0$. Далее, каждое из множеств $\Omega_1^1$ и $\Omega_1^2$ делим на два подмножества аналогично тому, как это делалось для $\Omega_0$. В результате, $\Omega_1^1$ разобьется на четыре подмножества $\Omega_2^1, \Omega_2^2, \overline{\Omega}_2^1, \overline{\Omega}_2^2$, такие, что $f(\tau) = 1$ при $\tau \in \overline{\Omega}_2^1$, $f(\tau) = 0$ при $\tau \in \overline{\Omega}_2^2$, т.е. вопрос о существовании и построении допустимого расписания для подмножеств $\overline{\Omega}_2^1$ и $\overline{\Omega}_2^2$ будет решен, а для $\Omega_2^1$ *и* $\Omega_2^2$ останется открытым.

Аналогично, $\Omega_1^2$ также разобьется на четыре подмножества $\Omega_2^3, \Omega_2^4, \overline{\Omega}_2^3, \overline{\Omega}_2^4$, такие, что для $\overline{\Omega}_2^3$ и $\overline{\Omega}_2^4$ вопрос о существовании и построении допустимого расписания будет решен, а для $\Omega_2^3$ и $\Omega_2^4$ останется открытым. При этом $|\Omega_2^1| + |\Omega_2^2| + |\Omega_2^3| + |\Omega_2^4| = \frac{1}{2}(|\Omega_1^1| + |\Omega_1^2|)$, т.е. число векторов из $\Omega_0$, для которых вопрос о существовании допустимого расписания остается открытым, после каждой операции деления на подмножества сокращается вдвое. Продолжим этот процесс для подмножеств $\Omega_2^1, \Omega_2^2, \Omega_2^3, \Omega_2^4$ и далее для вновь образовавшихся подмножеств, для которых вопрос о существовании допустимого расписания остается открытым. Указанный процесс будет завершен, когда для каждого $\tau \in \Omega_0$ вопрос о существовании и построении допустимого расписания будет решен.

Таким образом, на каждом шаге указанного выше процесса деления множества $\Omega_0$ на подмножества число векторов $\tau \in \Omega_0$, для которых вопрос о существовании допустимого расписания остается открытым, сокращается вдвое, а число подмножеств, содержащих эти вектора, увеличивается вдвое. Для каждого такого подмножества вычислительная сложность дихотомической процедуры составляет $O(\log T)$.

Будем предполагать, что все вектора $\tau \in \Omega_0$ равновероятны. Пусть $S$ – это вычислительная сложность программы $P(\tau)$ построения допустимого расписания $R(\tau)$. Если фиксировать некоторое $r \in N$ и остановиться после выполнения $r$ шагов процедуры деления множества $\Omega_0$ на подмножества (включая расчет допустимого расписания, когда оно существует), то вычислительная сложность такой процедуры составит $O(2^r S \log T)$. При этом вероятность того, что до проведения эксперимента расписание не было построено и его необходимо будет находить в реальном масштабе временив, т.е. в ходе проведения эксперимента, составляет $\frac{1}{2^r}$. В то же время, вычислительная сложность процедуры построения допустимого расписания для всех векторов $\tau \in \Omega_0$ составляет $O(ST^n)$. Учитывая, что $T$ и $n$ могут достигать больших значений, можно подобрать такое $r_0$, при котором $2^{r_0} S \log T \ll ST^n$, а величина $\frac{1}{2^{r_0}}$ достаточно мала. Иными словами, при $r = r_0$ вычислительная сложность предложенной процедуры много меньше вычислительной сложности полного перебора векторов $\tau \in \Omega_0$. В то же время, вероятность того, что при проведении эксперимента возникнет необходимость в расчете допустимого расписания $R(\tau)$, достаточно мала.

## 4. Заключение

Исследована задача составления допустимого расписания для совокупности комплексов работ, требования на выполнение которых поступают в заданные моменты времени. Состав каждого комплекса и характеристики входящих в него работ становятся известными в момент поступления запроса. При выполнении заданий допускаются прерывания и переключения с одного исполнительного механизма на другой. Разработан полиномиальный алгоритм решения задачи, основанный на построении сетевой потоковой модели и поиске максимального потока. Исследована задача построения допустимого расписания для совокупности работ с нефиксированными длительностями. Разработан метод предварительного (до проведения эксперимента в реальном масштабе времени) расчета расписаний, вычислительная сложность которого существенно меньше вычислительной сложности полного перебора всех возможных векторов длительностей. При этом существует небольшая вероятность того, что возникнет необходимость в расчете допустимого расписания в реальном масштабе времени.

## Литература


1. *Танаев В.С., Гордон В.С., Шафранский Я.М.* Теория расписаний. Одностадийные системы. – М.: Наука, 1984. – 383 с.
2. *Brucker P.* Scheduling Algorithms. – Heidelberg: Springer, 2007. – 371 p.
3. *Лазарев А.А.* Теория расписаний. Оценка абсолютной погрешности и схема приближенного решения задач теории расписаний. – М.: МФТИ, 2008. – 222 с.
4. *Мищенко А.В., Кошелев П.С.* Оптимизация управления работами логистического проекта в условиях неопределенности // Известия РАН. Теория и системы управления. 2021. № 4. – С.123–134.
5. *Горский М.А., Мищенко А.В., Нестерович Л.Г., Халиков М.А.* Некоторые модификации целочисленных оптимизационных задач с учетом неопределенности и риска // Известия РАН. Теория и системы управления. 2022. № 5. – С.106–117.
6. *Глонина А.Б., Балашов В.В.* О корректности моделирования модульных вычислительных систем реального времени с помощью сетей временных автоматов // Моделирование и анализ информационных систем. 2018. Т. 25. № 2. – С. 174–192.
7. *Глонина А. Б.* Обобщенная модель функционирования модульных вычислительных систем реального времени для проверки допустимости конфигураций таких систем // Вестник ЮУрГУ. Сер. Вычисл. математика и информатика. 2017. Т. 6. № 4. – С. 43–59.
8. *Глонина А. Б.* Инструментальная система проверки выполнения ограничений реального времени для конфигураций модульных вычислительных систем // Вестник МГУ. Сер. 15. Вычисл. математика и кибернетика. 2020. № 3. – С. 16–29.
9. *Алифанов Д.В., Лебедев В.Н., Цурков В.И.* Оптимизация расписаний с логическими условиями предшествования // Известия РАН. ТиСУ. 2009. № 6. – С.88–93.
10. *Миронов А.А., Цурков В.И.* Минимакс в моделях транспортного типа с интегральными ограничениями // Известия РАН. ТиСУ. 2003. № 4. – С.69–81.
11. *Миронов А.А., Цурков В.И.* Минимакс при нелинейных транспортных ограничениях // ДАН. 2001. Т. 381. № 3. – С. 305–308.
12. *Филлипс Д., Гарсиа-Диас А.* Методы анализа сетей. – М.: Мир, 1984. – 496 с.
13. *Давыдов Э.Г.* Исследование операций. – М.: Высшая школа, 1990. – 384 с.
14. *Косоруков Е.О., Фуругян М.Г.* Некоторые алгоритмы распределения ресурсов в многопроцессорных системах // Вестник МГУ. Сер. 15. 2009. № 4. – С. 34–37.
15. *Кононов Д.А., Фуругян М.Г.* Распределение неоднородного комплекса ресурсов при региональном управлении // Управление развитием крупномасштабных систем (MLSD’2021): Труды тринадцатой междунар. конф. – М.: ИПУ РАН, 2020. – С. 1298–1305.
16. *Кононов Д.А., Фуругян М.Г.* Оптимизация использования неоднородного комплекса ресурсов при региональном планировании // Управление развитием крупномасштабных систем (MLSD’2021): Труды четырнадцатой междунар. конф. – М.: ИПУ РАН, 2021. – С. 1231–1237.
17. *Фуругян М.Г.* Планирование вычислений в многопроцессорных системах с несколькими типами дополнительных ресурсов и произвольными процессорами // Вестник МГУ. Сер. 15. 1917. № 3. – С. 38–45.
18. *Kononov D., Furugyan M.* Control of a Complex of Works in Multiprocessor Real-time ACS // Proceedings of the 1st International Conference on Control Systems, Mathematical Modeling, Automation and Energy Efficiency (SUMMA2019, Lipetsk). – Lipetsk: IEEE, 2019. URL: https://ieeexplore.ieee.org/document/8947570 (дата обращения: 27.04.2023).
19. *Kononov D., Furugyan M.* Distribution of Non-Uniform Complex of Resources in Production Systems // IFAC-PapersOnLine. – Nantes, France: Elsevier, 2022. – Volume 55, Issue 10. – P. 2138–2143.
20. *Kononov D., Furugyan M.* Effective Management of Regional Projects Resources // Proceedings of the 12th International Conference "Management of Large-Scale System Development" (MLSD). – M.: IEEE, 2019. – P. 1–5.

URL: https://ieeexplore.ieee.org/document/8910972 (дата обращения: 20.04.2023).
21. *Kononov D., Furugyan M.* Ensuring the Security of the Operation of Complex Systems: New Models and Algorithms // IFAC-PapersOnLine. – Moscow: Elsevier Ltd., 2021. – Vol. 54, Iss. 13.– P. 123–128.
22. *Kononov D., Furugyan M.* Optimization of Work Planning in the Production of Innovations // Proceedings of the 4th International Conference on Control Systems, Mathematical Modeling, Automation and Energy Efficiency (SUMMA). – Lipetsk: IEEE, 2022. – P. 592–596.
23. *Kononov D., Furugyan M.* Planning a Complex of Works with Heterogeneous Resources under Uncertainty // Proceedings of the 15th International Conference Management of Large-Scale System Development (MLSD). – Moscow: IEEE, 2022. – P. 1–5. URL: https://ieeexplore.ieee.org/document/9934381 (дата обращения: 19.04.2023).
24. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К.* Алгоритмы. Построение и анализ (второе издание.) – М.: Вильямс, 2005. – 1296 с.
25. *Фуругян М.Г.* Некоторые алгоритмы решения минимаксной задачи составления многопроцессорного расписания // Известия РАН. ТиСУ. 2014. №2. – С. 50–56.
26. *Federgruen A., Groenevel H.* Preemptive Scheduling of Uniform Machines by Ordinary Network Flow Technique // Management Science. – 1986. – Vol. 32. № 3. – P. 341–349.