УДК 004: 519.854

ГРАДИЕНТНЫЙ МЕТОД СОСТАВЛЕНИЯ ДИНАМИЧЕСКИХ РАСПИСАНИЙ ДЛЯ КОНВЕЙЕРНЫХ СИСТЕМ, УЧИТЫВАЮЩИЙ ОТКАЗЫ СЕГМЕНТОВ

К. В. Кротов, Т. Ю. Кротова

Севастопольский государственный университет

Поступила в редакцию 06.05.2015 г.

Аннотация. В работе обосновываются модель составления расписаний обработки данных в конвейерных системах и метод построения динамических расписаний, учитывающий отказы сегментов конвейера и основывающийся на жадных стратегиях.
Ключевые слова: многостадийная конвейерная система, расписания выполнения программ обработки данных, жадный алгоритм, динамические расписания, отказ сегмента.
Annotation. The model of creating schedules processing of data in the conveyor system and method for constructing dynamic schedules, which were taking into account the failure of the device and based on the greedy strategy, were found in work.
Keywords: multi-stage conveyor system, schedule of execution of data processing programs, the gradient method, the segment fails, the greedy algorithm.

ВВЕДЕНИЕ

Конвейеризация программ предполагает их разделение на фрагменты, каждый из которых закреплен для выполнения за соответствующим сегментом конвейера. Введем в рассмотрение обозначения: i – идентификатор типа обрабатываемых данных, n – количество типов данных ($i=\overline{1,n}$), n^i – количество данных i-го типа. Данные i-го типа обрабатываются соответствующей им программой, тогда индекс i соответствует программе, выполняемой в составе конвейера, обрабатывающей данные i-го типа. Если количество данных i-го типа – n^i элементов, то обрабатывающая эти данные программа должна быть выполнена в конвейерной системе n^i раз. Цель функционирования конвейерной системы в этом случае состоит в обработке поступающих на ее вход данных выполняющимися в системе конвейеризированными программами. Все выполняющие обработку данных конвейеризированные программы находятся в оперативной памяти каждого из сегментов конвейера. Тогда управление вычислительным процессом в конвейерных системах предполагает определение порядка запуска программ обработки данных на выполнение. Так как объемы вычислений на каждом сегменте различны, являются различными длительности выполнения программ на соответствующих сегментах, тогда может быть сформировано расписание выполнения конвейеризированных программ обработки соответствующих данных, представляющее собой порядок запуска программ на выполнение.

Постановка задачи управления вычислительным процессом предполагает, что при $n^i=1$ ($i=\overline{1,n}$) реализуется обработка единичных данных (однократный запуск на выполнение программ), для которых должно быть сформировано расписание их обработки. В тоже время ход вычислительного процесса подвержен возмущающим воздействиям, к которым могут быть отнесены отказы сегментов конвейера. Отказы сегмента конвейера могут быть связаны с «зависанием» программы, обрабатывающей данные, операционной системы (при этом выполнение программы также не может быть продолжено), возможны отказы аппаратного обеспечения.

Устранение влияния возмущений (в частности, отказов сегментов) на ход вычислительного процесса (на выполнение сформированного статического расписания) реализуется путем построения динамических расписаний, учитывающих соответствующий вид возмущений.

АНАЛИЗ ПУБЛИКАЦИЙ

Современные методы теории расписаний позволяют формировать статические расписания обработки единичных данных разных типов при заданном количестве приборов в многостадийных обрабатывающих системах (в частности, в конвейерных системах) с использованием различных критериев. В работе [1] выполнен анализ основных подходов к решению задач формирования расписаний обработки данных. Для решения задач теории расписаний развиваются точные методы (ветвей и границ, ветвей и отсечений), приближенные методы и методы локальной оптимизации (метод отжига, генетические алгоритмы и т. д.). Реализация методов предполагает использование различных видов критериев оптимизации, учет ограничений на директивные сроки окончания обслуживания, возможность задания как фиксированного маршрута обработки данных, так и произвольного маршрута. Развитие методов теории расписаний связано с решением задач групповой обработки (обработки партий), однако методы построения расписаний при групповой обработке предполагают формирование фиксированных партий данных (включающих данные одного типа) при обработке на ограниченном количестве сегментов. Динамические свойства расписаний, связанные с различными событиями, происходящими в системе, ни в одной из анализируемых работ не рассматриваются. Таким образом, решение задачи построения динамических расписаний при учете возмущающих воздействий, изменяющих запланированный ход вычислительного процесса, является актуальным.

ПОСТАНОВКА ЦЕЛИ И ЗАДАЧ НАУЧНОГО ИССЛЕДОВАНИЯ

Цель работы состоит в совершенствовании методов построения расписаний обработки единичных данных в конвейерных системах. Совершенствование методов построения расписаний связано с применением «жадного» подхода в дискретной оптимизации. Это позволит реализовать учет возмущающих воздействий на ход вычислительного процесса и сформировать методы построения динамических расписаний выполнения программ обработки данных. При этом возмущающим воздействием, которое учитывается при формировании динамического расписания, является отказ сегмента конвейера.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Обозначим через d_i момент времени поступления в систему данных i-го типа ($i=\overline{1,n}$). Для всех типов данных моменты их поступления на обработку одинаковы, при этом $d_i = 0$. Обозначим через l индекс сегмента конвейерной системы, осуществляющего выполнение l-й части программы, L – общее количество сегментов конвейера ($l=\overline{1,L}$). Каждым сегментом конвейера выполняются вычисления, соответствующие назначенной ему части программы. Дисциплина обслуживания в системе предполагает прохождение данными всех сегментов конвейера, при этом если l-й сегмент приступил к обработке данных i-го типа, обработка не может быть прервана обработкой других данных. Выполнение на каждом l-м сегменте назначенной ему части i-го программы характеризуется параметром длительности обработки. Так как в системе выполняется обработка единичных данных ($n^i = 1$ при $i=\overline{1,n}$), для каждой из n обрабатывающих программ будет выполнен однократный ее запуск на выполнение. Отказ l-го сегмента конвейера прерывает обработку данных некоторого i-го типа, после чего сегмент простаивает, находясь в состоянии восстановления после отказа. Переход отказавшего сегмента в работоспособное состояние обеспечивает продолжение обработки данных на нем.

Расписание обработки данных (запуска на выполнение программ) обозначим как π. Расписание обработки данных π – это совокупность последовательностей π^l запуска данных на обработку на каждом l-м сегменте конвейера ($l=\overline{1,L}$), оно имеет вид: $\pi=\{\pi^1,\pi,^2\pi^3,...,\pi^L\}$. Так как виды последовательностей π^l ($l=\overline{1,L}$) различны, то для их формализации в рассмотрение введены матрицы P^l ($l=\overline{1,L}$) порядков обработки данных в системе (порядков запуска обрабатывающих программ на выполнение). Элемент $p_{ij}^l=1$, если данные i-го типа занимают в последовательности π^l j-ю позицию, $p_{ij}^l=0$ в противном случае, размеры матриц $n\times n$, где n – количество типов данных и количество позиций этих данных в последовательностях π^l. Запланированный ход вычислительного процесса реализуется в соответствии с сформированным статическим расписанием. Формирование динамических расписаний предполагает перестроение полученных ранее статических расписаний с учетом номера отказавшего сегмента, момента времени наступления отказа и длительности восстановления сегмента. Возмущающее воздействие в форме отказа сегмента конвейера влияет на запланированный ход вычислительного процесса, тогда первоначально должно быть сформировано статическое расписание обработки данных, которое в дальнейшем модифицируется с учетом отказа. Итогом модификации является сформированное динамическое расписание. По этой причине первоначально должен быть сформулирован метод формирования статических расписаний для n типов обрабатываемых в системе данных, затем обоснованы особенности формирования последовательностей π^l ($l=\overline{1,L}$) при построении динамических расписаний с учетом отказа и восстановления сегмента конвейера. Эффективность формируемых расписаний характеризуется значениями критерия, формулируемого на основе модели вычислительного процесса. При построении статических и динамических расписаний используется одинаковый вид критерия эффективности.

Для формализации модели вычислительного процесса обработки данных конвейерной системой в рассмотрение введены обозначения: t_i^l – длительность интервала обработки данных i-го типа на l-м сегменте конвейера ($l=\overline{1,L}$); (t_{ji}^{0l}) – матрица моментов времени начала обработки данных i-ых типов, занимающих в π^l j-е позиции, $\overline{t_{ji}^l}$ –момент времени окончания обработки данных i-го типа, занимающих в π^l j-ю позицию. Перед началом обработки данных все выполняющиеся программы загружены в оперативную память сегментов конвейера и непосредственно при поступлении данных в момент времени $d_i=0$ начинается их обработка, тогда длительности первоначальной наладки сегментов конвейера на обработку данных i-ых типов не учитываются. Так как в системе выполняется обработка единичных данных, тогда длительности переналадки сегментов с обработки данных i-го типа на обработку данных k-го типа могут быть включены в интервалы t_i^l обработки данных соответствующих i-ых типов ($i=\overline{1,n}$). Значения элементов матриц (t_{ji}^{01}) ($l=\overline{1,L}$) определяются в соответствии с видом матриц P^l ($l=\overline{1,L}$) следующим образом: $t_{ji}^{0l}\neq 0$ для того элемента матрицы (t_{ji}^{0l}), который соответствует $p_{ij}^l=1$. В случае, если $p_{ij}^l=0$, то элемент t_{ji}^{0l} матрицы (t_{ji}^{0l}) равен 0 ($t_{ji}^{0l}=0$). Для первого сегмента конвейера элементы матрицы (t_{ji}^{01}) определяются с учетом матрицы $\left(\boldsymbol{P}\right)^1$ следующим образом:

$$t_{1i}^{01}=0;\ t_{2i}^{01}=\sum_{h=1}^{n}t_h^1\cdot p_{h1}^1;\ t_{3i}^{01}=\sum_{j=1}^{2}\sum_{h=1}^{n}t_h^1\cdot p_{h,j}^1;$$

$$t_{4i}^{01}=\sum_{j=1}^{3}\sum_{h=1}^{n}t_h^1\cdot p_{h,j}^1 \text{ и т. д.}$$

В общем виде выражения для определения t_{ji}^{01} имеют следующую форму:

$$t_{ki}^{01}=\sum_{j=1}^{k-1}\sum_{h=1}^{n}t_h^1\cdot p_{h,j}^1,\ \text{при } k>1, \qquad (1)$$

где k – номер позиции данных i-го типа в последовательности π^1.

Для l-го сегмента (при $l\neq 1$) элементы матрицы (t_{ji}^{0l}) определяются выражениями вида:

а) первая строка (первая позиция данных i-го типа, $j=1$):

$$t_{1,i}^{0l}=\sum_{h=1}^{n}p_{ih}^{l-1}\cdot(t_{h,i}^{0l-1}+t_i^{l-1}), \qquad (2)$$

где индекс i типа данных определяется по матрице $\left(\boldsymbol{P}\right)^l$ ($l=\overline{1,L}$) как занимающих первую позицию в последовательности π^l;

б) j-я строка ($j\neq 1$) в матрице $\left(\boldsymbol{t_{ji}^{0l}}\right)$ ($l=\overline{2,L}$) для данных i-го типа:

$$t_{ji}^{0l}=\max\left\{\sum_{h=1}^{n}p_{i,h}^{l-1}\cdot(t_{h,i}^{0l-1}+t_i^{l-1});\ \sum_{h=1}^{n}(t_{j-1,h}^{0l}+t_h^l)\cdot p_{h,j-1}^l\right\}. \quad (3)$$

Сформированные выражения (1)–(3) являются моделью вычислительного процесса обработки единичных данных, т. е. позволяют определять его временные характеристики. Формируемые значения характеристик вычислительного процесса являются различными в зависимости от видов сформированных решений, которые представляются в форме: $[P^l,(t_{ji}^{0l})\,|\,l=\overline{1,L}]$. Для анализа эффективности получаемых решений должен быть сформирован критерий, учитывающий значения характеристик вычислительного процесса.

Особенностями алгоритма метода определения порядка обработки данных, реализующего «жадный» подход к оптимизации решений ([1]), являются: 1) добавление в конец сформированных на предыдущем ($(s-1)$-ом) шаге алгоритма последовательностей $\pi^l(s-1)$ программы i-го типа обработки данных; 2) определение позиции рассматриваемой программы i-го типа в вновь формируемых последовательностях $\pi^l(s)$ на текущем (s-ом) шаге алгоритма; 3) вычисление и анализ градиентов целевой функции после реализации каждого шага алгоритма, связанного с изменением положения в одной последовательности π^l рассматриваемой программы на одну позицию ближе к началу; 4) если изменение положения программы i-го типа в каждой последовательности π^l ($l=\overline{1,L}$) на одну позицию ближе к ее началу не приводит к уменьшению значения целевой функции, тогда реализуется изменение положения программы одновременно в двух, трех последовательностях π^l и т. д. Для текущего решения реализуется переход к более эффективному решению в рамках окрестностей с различными метриками. В [1] сформулированы особенности реализации «жадного» выбора при построении расписаний обработки данных и доказана возможность получения локально эффективного решения задачи.

При учете того, что на каждом s-ом шаге алгоритма в полученные ранее локально эффективные последовательности π^l добавляется одна программа обработки данных i-го типа, оценка эффективности видов последовательностей реализуется для текущего количества программ в них. Для обоснования вида критерия эффективности формируемого решения выполнены следующие рассуждения:

1) при $\sum_{i=1}^{n}t_{ji}^{0l}\cdot p_{ij}^l>\sum_{i=1}^{n}(t_{j-1,i}^{0l}+t_i^l)\cdot p_{i,j-1}^l$ l-й сегмент конвейера ожидает готовности для обработки данных в j-ой позиции в π^l после их обработки в $(j-1)$-ой позиции; длительность простоев l-го сегмента конвейера в ожидании данных, занимающих j-ю позицию в π^l определяется следующим образом: $\sum_{i=1}^{n}t_{j,i}^{0l}\cdot p_{i,j}^l-\sum_{i=1}^{n}(t_{j-1,i}^{0l}+t_i^l)\cdot p_{i,j-1}^l$, длительность простоев l-го сегмента конвейера при выполнении текущего количества программ обработки данных, находящихся в π^l, определяется выражением:
$\sum_{j=2}^{n}(\sum_{i=1}^{n}t_{j,i}^{0l}\cdot p_{i,j}^l-\sum_{i=1}^{n}(t_{j-1,i}^{0l}+t_i^l)\cdot p_{i,j-1}^l)$;

2) для первого сегмента конвейера простои в ожидании готовности данных отсутствуют, суммарное время простоев l-ых сегментов $(l=\overline{2,L})$, связанных с ожиданием готовности данных, определено выражением вида: $\sum_{l=2}^{L}\sum_{j=2}^{n}(\sum_{i=1}^{n}t_{j,i}^{0l}\cdot p_{i,j}^l-\sum_{i=1}^{n}(t_{j-1,i}^{0l}+t_i^l)\cdot p_{i,j-1}^l)$;

3) если данные занимают первую позицию в последовательностях π^l $(l=\overline{2,L})$, тогда простои сегментов связаны также с ожиданием поступления этих данных на обработку; для l-го сегмента (при условии, что $l\neq 1$) время ожидания готовности данных для обработки в первой позиции π^l $(l=\overline{2,L})$ определяется выражением вида: $\sum_{i=1}^{n}t_{1,i}^{0l}\cdot p_{i1}^l$; т. к. $t_{1i}^{01}=0$, тогда простои l-ых сегментов конвейера $(l=\overline{2,L})$ будут определены выражением $\sum_{l=2}^{L}\sum_{i=1}^{n}t_{1,i}^{0l}\cdot p_{i1}^l$.

В соответствии с выполненными рассуждениями вид критерия эффективности формируемых решений по порядку обработки данных в π^1 ($l=\overline{1,L}$) следующий:

$$f=\sum_{l=2}^{L}\sum_{i=1}^{n}t_{1,i}^{0l}\cdot p_{i,1}^{l}+ \\ +\sum_{l=2}^{L}\sum_{j=2}^{n}\left(\sum_{i=1}^{n}t_{ji}^{0l}\cdot p_{ij}^{l}-\sum_{i=1}^{n}(t_{j-1,i}^{0l}+t_{i}^{l})\cdot p_{i,j-1}^{l}\right). \quad (4)$$

Предложенный в [1] метод построения статических расписаний обработки данных в конвейерных системах является основой для обоснования метода построения динамических расписаний, учитывающего отказы сегментов конвейера. При этом первоначально реализуется формирование статического расписания обработки данных, которое после фиксации отказа сегмента конвейера модифицируется с использованием формулируемого метода построения динамических расписаний.

Формулировка задачи построения динамического расписания предполагает, что обработка данных на некотором l-ом сегменте прерывается в связи с его отказом и l-ый сегмент простаивает в течение интервала времени восстановления. Для удобства рассуждений отказ l-го сегмента проинтерпретирован как поступление на него для обработки данных $i_{вос}$ (восстановление сегмента после отказа), имеющих наивысший приоритет обслуживания ($R_{i_{вос}}>R_i$, где R_i – приоритеты данных i-ых типов ($i=\overline{1,n}$), которые являются одинаковыми). Тогда момент времени наступления отказа l-го сегмента сопоставим с моментом времени поступления данных $i_{вос}$ на обработку на этот сегмент и обозначим его как $t_{i_{вос}}^{0l}$, длительность восстановления l-го сегмента после отказа обозначим как $t_{i_{вос}}^{l}$ (соответствует интервалу времени «обработки» данных $i_{вос}$ на l-ом сегменте). Таким образом, момент времени реализации отказа l-го сегмента отождествлен с моментом времени $t_{i_{вос}}^{0l}$ начала восстановления l-го сегмента после отказа.

Для реализации дальнейших рассуждений выполнен переход к обозначениям типа данных и их позиций в виде i_j^l, где j – некоторая позиция данных i-го типа в рассматриваемой последовательности π^l l-го сегмента. Рассуждения, касающиеся учета влияния отказа сегмента конвейера на расписание обработки данных, формулируются с учетом рис. 1.

Так как при отказе l-го сегмента происходит прерывание обработки данных i_q^l (данных i-го типа в q-ой позиции в π^l), то обработка этих данных может быть представлена состоящей из двух стадий: $(t_{i_q}^l)_1$ и $(t_{i_q}^l)_2$, где $(t_{i_q}^l)_1$ – длительность стадии обработки, предшествующей отказу l-го сегмента; $(t_{i_q}^l)_2$ – длительность оставшейся обработки данных i_q^l. В силу того, что момент времени $t_{i_{вос}}^{0l}$ отказа сегмента является случайным, то случайными будут являться длительности $(t_{i_q}^l)_1$ и $(t_{i_q}^l)_2$ стадий обработки данных i_q^l. Длительность заключительной стадии обработки данных i_q^l будет определена следующим образом: $(t_{i_q}^l)_2=t_{i_q}^l-(t_{i_q}^l)_1$, при условии, что $(t_{i_q}^l)_1=t_{i_{вос}}^{0l}$, где $(t_{i_q}^l)_1$ – момент окончания реализации первой стадии обработки данных i_q^l. Условие

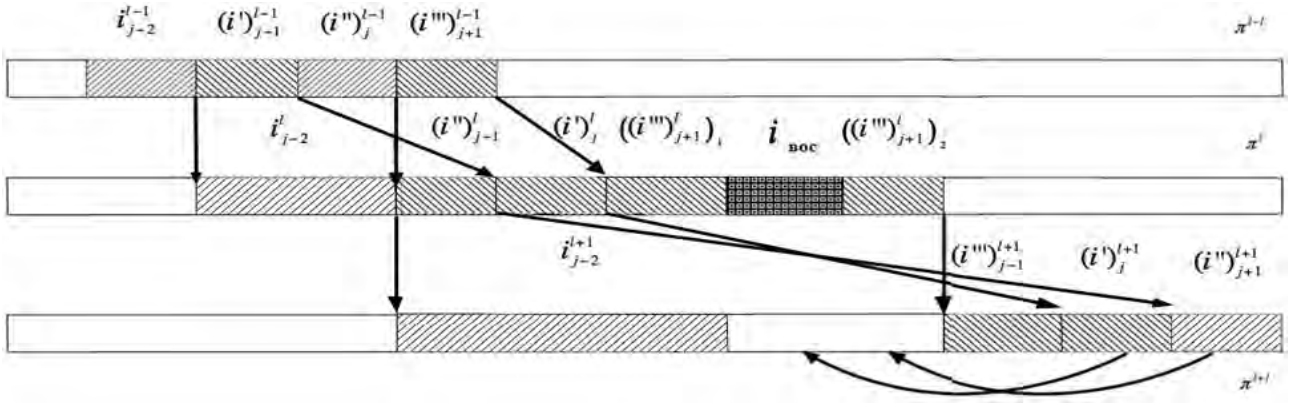

Рис. 1. Вид последовательностей обработки данных с учетом отказа l-го сегмента конвейера

$(t_{i_q}^{0l})_2 = \overline{t_{i_{вос}}^{l}}$ является обязательным, начало реализации второй стадии обработки данных i_q^l на l-ом сегменте совпадает с окончанием восстановления этого сегмента. Восстановление сегмента после отказа в течение интервала $t_{i_{вос}}^{l}$ приводит к изменению параметров $t_{i_j}^{0l}$, $t_{i_j}^{l}$ (в введенных обозначениях t_{ji}^{0l}, $\overline{t_{ji}^{l}}$). При реализации отказа возможны простои $(l+1)$-го, $(l+2)$-го и т. д. сегментов в системе, связанные с восстановлением l-го сегмента и отсутствием данных для обработки на $(l+1)$-ом и $(l+2)$-ом и т. д. сегментах. Сформулированное утверждение комментируют рис. 1. На рис. 1 в последовательность π^l введены данные $i_{вос}$, имитирующие восстановление l-го сегмента после отказа, прерывающее обработку данных $(i''')_{j+1}^{l}$. Восстановление l-го сегмента, обозначенное как $i_{вос}$, введенное в последовательность π^l, вызывает простой $(l+1)$-го сегмента по причине неготовности данных $(i''')_{j+1}^{l}$ к обслуживанию (при этом предполагается реализация порядка обработки данных на $(l+1)$-ом сегменте, соответствующего последовательности π^{l+1} в статическом расписании). В этом случае возникает задача изменения вида последовательностей π^l ($l=\overline{1,L}$) для данных, еще не обработанных к моменту времени наступления отказа l-го сегмента, с учетом длительности его восстановления, т. е. требуется сформировать динамическое расписание, предполагающего изменение порядка данных в последовательностях π^l.

При формировании динамического расписания могут быть выделены данные, порядок которых в последовательностях π^l ($l=\overline{1,L}$) не может быть изменен, и данные, порядок обработки которых может быть определен динамическим расписанием с учетом длительности восстановления отказавшего сегмента. Для l-го сегмента конвейера ($l=\overline{1,L}$) введена в рассмотрение последовательность π_1^l с неизменным порядком обработки данных и последовательность π_2^l, порядок обработки данных в которой может быть изменен при учете времени восстановления отказавшего сегмента. Последовательности π_1^l и π_2^l образуют последовательность π^l обработки данных на l-ом сегменте ($\pi^l = \pi_1^l \cup \pi_2^l$). Вид последовательности π^l при определении в ней последовательностей π_1^l и π_2^l:

$$\pi^l = \{i_1^l, i_2^l, \ldots, i_q^l, i_{q+1}^l, i_{q+2}^l, \ldots, i_n^l\}, \qquad (5)$$

где данные $i_1^l, i_2^l, \ldots, i_q^l$ входят в последовательность π_1^l, а данные i_{q+1}^l, $i_{q+2}^l, \ldots$, i_n^l – в последовательность π_2^l. Данные i_q^l являются правой границей последовательности π_1^l; для данных $i_{q+1}^l, i_{q+2}^l, ..., i_n^l$, входящих в последовательность π_2^l, может быть изменен порядок их обработки на l-ом сегменте с учетом отказа и восстановления одного из сегментов конвейера. Для отказавшего сегмента введем в рассмотрение обозначение $\hat{l}$ (среди всех сегментов происходит отказ $\hat{l}$-ого сегмента). Вид последовательности $\pi^{\hat{l}}$, в которой должны быть размещены данные $i_{вос}$, с учетом двух стадий обработки данных i_q^l, выполнение операций с которыми прерывается отказом сегмента $\hat{l}$, следующий:

$$\pi^{\hat{l}} = \left\{i_1^{\hat{l}}, i_2^{\hat{l}}, \ldots, (i_q^{\hat{l}})_1, i_{вос}^{\hat{l}}, (i_q^{\hat{l}})_2, \ldots, i_n^{\hat{l}}\right\}. \qquad (6)$$

Здесь первая стадия обработки данных $i_q^{\hat{l}}$, обозначенная как $(i_q^{\hat{l}})_1$, совместно с восстановлением сегмента после отказа, обозначенного как $i_{вос}^{\hat{l}}$, и второй стадией обработки данных $i_q^{\hat{l}}$, обозначенной как $(i_q^{\hat{l}})_2$, являются правой границей последовательности $\pi_1^{\hat{l}}$, в которой порядок обработки данных не может быть изменен. Для данных i_j^l (j – текущая позиция данных, $j=\overline{1,q-1}$, $l=\overline{1,L}$, $l \neq \hat{l}$) и данных i_q^l, являющихся последним элементом в π_1^l, сформулированы условия, определяющие их принадлежность π_1^l:

а) для данных i_j^l: $t_{i_j}^{0l} < t_{i_{вос}}^{0\hat{l}}$, $\overline{t_{i_j}^{l}} < t_{i_{вос}}^{0\hat{l}}$; (7)

б) для данных i_q^l: $t_{i_q}^{0l} < t_{i_{вос}}^{0\hat{l}}$, $t_{i_{вос}}^{0\hat{l}} \leq \overline{t_{i_q}^{l}}$. (8)

Для данных i_j^l $(j=\overline{q+1,n})$ условие их принадлежности последовательности π_2^l имеет вид: $t_{i_{вос}}^{0\hat{l}} \leq t_{i_j}^{0l}$. То есть данные $i_j^l \in \pi_2^l$ $(j=\overline{q+1,n})$ являются еще не обработанными сегментом l ($l=\overline{1,L}$, $l \neq \hat{l}$) к моменту времени $t_{i_{вос}}^{0\hat{l}}$ отказа $\hat{l}$-го сегмента. Аналогичные (7), (8) условия принадлежности данных $i_j^{\hat{l}}$ $(j=\overline{1,q-1})$ последовательности $\pi_1^{\hat{l}}$ для отказавшего сегмента имеют вид:

а) для данных $i_j^{\hat{l}}$: $t_{i_j}^{0l} < t_{i_{вос}}^{0\hat{l}}$, $\overline{t_{i_q}^{\hat{l}}} < t_{i_{вос}}^{0\hat{l}}$; (9)

б) для данных $i_q^{\hat{l}}$: $t_{i_q}^{0\hat{l}} < t_{i_{вос}}^{0\hat{l}}$, $t_{i_{вос}}^{0\hat{l}} \le \overline{t_{i_q}^{\hat{l}}}$. (10)

В случае если $t_{i_{вос}}^{0\hat{l}} = \overline{t_{i_q}^{\hat{l}}}$, то длительность второй стадии обработки данных $i_q^{\hat{l}}$ является нулевой (т. е. $(t_{i_q}^{\hat{l}})_2 = 0$) и данные $i_q^{\hat{l}}$ целиком принадлежат последовательности $\pi_1^{\hat{l}}$. Если $t_{i_{вос}}^{0\hat{l}} < t_{i_q}^{\hat{l}}$, то в последовательности $\pi_1^{\hat{l}}$ местоположение второй стадии $(i_q^{\hat{l}})_2$ обработки данных $i_q^{\hat{l}}$ следует непосредственно за $i_{вос}$ и не может быть изменено. Поэтому q-я позиция данных $i_q^{\hat{l}}$ является правой границей последовательности $\pi_1^{\hat{l}}$ и длительность обработки этих данных (при расчете значений элементов матрицы (t_{ji}^{0l})) модифицируется следующим образом: $(t_{i_q}^{\hat{l}})' = t_{i_q}^{\hat{l}} + t_{i_{вос}}^{\hat{l}}$. В соответствии с условиями принадлежности данных i_j^l, $i_j^{\hat{l}}$ последовательностям π_1^l, $\pi_1^{\hat{l}}$ ($j = \overline{1,q}$) вида (7)–(10) введены условия для определения принадлежности данных i_j^l, $i_j^{\hat{l}}$ ($j = \overline{q+1,n}$) последовательностям π_2^l и $\pi_2^{\hat{l}}$. Сформулированные условия для i_j^l, $i_j^{\hat{l}}$ имеют вид:

а) для данных i_j^l: $t_{i_{вос}}^{0\hat{l}} < t_{i_j}^{0l}$ ($j = \overline{q+1,n}$, $l = \overline{1,L}$, $l \ne \hat{l}$); (11)

б) для данных $i_j^{\hat{l}}$: $t_{i_{вос}}^{0\hat{l}} < t_{i_j}^{0\hat{l}}$ ($j = \overline{q+1,n}$).(12)

Условия (7)–(12) позволяют определить составы последовательностей π_1^l и π_2^l ($l = \overline{1,L}$) в статическом расписании, т. е. определить данные i_j^l, входящие в последовательность π_2^l, порядок обработки которых на сегментах конвейера изменяется. В соответствии с условиями (7)–(12) для каждого из сегментов может быть сформировано множество N^l ($l = \overline{1,L}$) данных, порядок которых в последовательностях π_2^l будет переопределяться. Формируемые с учетом (7)–(12) множества N^l ($l = \overline{1,L}$) имеют вид:

а) для l-го сегмента: $N^l = \{i_{q+1}^l, i_{q+2}^l, ..., i_n^l\}$;

б) для $\hat{l}$-го сегмента: $N^{\hat{l}} = \{i_{q+1}^{\hat{l}}, i_{q+2}^{\hat{l}}, ..., i_n^{\hat{l}}\}$.

Для реализации алгоритма построения динамических расписаний для данных, которые не были обработаны к моменту времени $t_{i_{вос}}^{0\hat{l}}$ наступления отказа $\hat{l}$-го сегмента конвейера в рассмотрение введено обозначение $N_{пр}$ (not processed), формируемое следующим образом: $N_{пр} = \bigcup_{l=1}^{L} N^l$. Введем в рассмотрение обозначение i' типа данных, которые будут добавляться в последовательности π^l ($l = \overline{1,L}$) на s-ом текущем шаге алгоритма при построении динамического расписания. Для определения порядка действий алгоритма градиентного метода составления динамических расписаний при зафиксированном отказе $\hat{l}$-го сегмента необходимо предварительно ввести в рассмотрение способ формирования последовательностей π_2^l ($l = \overline{1,L}$). Расчет значений критерия вида (4) возможен в случае, когда количество данных в последовательностях π^l ($l = \overline{1,L}$) является одинаковым, при этом в π^l размещены данные одинаковых типов. Если проинтерпретировать π^l ($l = \overline{1,L}$) как упорядоченные множества данных, то для расчета значений критерия (4) необходимо, чтобы $|\pi^l| = |\pi^{l'}|$ ($l \ne l'$). Поэтому способ формирования последовательностей π_2^l ($l = \overline{1,L}$) предусматривает реализацию двух последовательных этапов:

1) размещение в последовательностях π_2^l ($l = \overline{1,L}$) данных $i' \in N_{пр}$ таких, что $i' \notin \pi_1^l$ и $i' \in \pi_1^{l'}$ (данные i' не входят в последовательность π_1^l l-го сегмента конвейера, но входят в последовательность $\pi_1^{l'}$ другого l'-го сегмента, $l \ne l'$) и определение в π_2^l эффективного положения этих данных; на первом этапе выполняется определение эффективного местоположения в π_2^l (в π^l) всех данных i', для которых перед началом построения динамического расписания выполняется условие $i \in N_{пр} \cap (\bigcup_{l=1}^{L} \pi_1^l)$;

2) размещение в последовательностях π_2^l данных i-ых типов, для которых перед началом построения расписания выполняется условие $(i \in N_{пр}) \& (i \notin (\bigcup_{l=1}^{L} \pi_1^l))$, и определение эффективного местоположения этих данных в последовательностях π_2^l (т. е. размещение в последовательностях π_2^l данных i-ых типов, которые перед началом построения расписания не входили в последовательности π_1^l ни на одном l-ом сегменте ($l = \overline{1,L}$)).

Для реализации первого этапа определения эффективных порядков обработки данных в π_2^l введем в рассмотрение множество $N'_{пр}$ типов i данных, для которых $i \in N_{пр} \cap (\bigcup_{l=1}^{L} \pi_1^l)$; Если для данных i-го типа введенное условие выполняется, тогда $N'_{пр} = N'_{пр} \cup \{i\}$. Формирование начального решения по порядкам обработки данных в последовательностях π_2^l $(l = \overline{1, L})$ предполагает добавление в них данных i, для которых выполняются условия: $i \in N'_{пр}$ и $i \notin \pi_1^l$ $(l = \overline{1, L})$ (в π_2^l на l-ом сегменте добавляются данные i-го типа, которые не входят в последовательность π_1^l на этом же сегменте, но содержатся в последовательности $\pi_1^{l'}$ некоторого l'-го сегмента $(l \neq l')$). В силу выполненных рассуждений последовательность шагов алгоритма, реализующего формирование начального решения по составам последовательностей π_2^l $(l = \overline{1, L})$ для данных $i \in N'_{пр}$, имеет вид (на текущем шаге рассматривается тип данных i', множество $N'_{пр}$ предварительно упорядочивается по возрастанию идентификаторов i типов данных в нем):

1) из множества $N'_{пр}$ выбирается тип данных i': $i' = \min\{i \mid i \in N'_{пр}\}$ (в итоге определяется необходимость добавления данных i'-го типа в последовательности π_2^l);

2) для данных i'-го типа выполняется проверка условия $i' \in \pi_1^l$ $(l = \overline{1, L})$, в случае его выполнения данные i'-го типа не могут быть добавлены в π_2^l; если $i' \notin \pi_1^l$ тогда данные i' добавляются в последовательность π_2^l: $\pi_2^l = \pi_2^l \cup \{i'\}$;

3) после добавления типа данных i' в последовательности π_2^l этот тип данных удаляется из множеств $N'_{пр}$ и $N_{пр}$: $N'_{пр} = N'_{пр} \setminus \{i'\}$; $N_{пр} = N_{пр} \setminus \{i'\}$; проверка условия $N'_{пр} \neq \varnothing$; в случае выполнения условия реализуется переход на шаг 1; при не выполнении условия все данные i $(i \in N'_{пр})$ размещены в π_2^l, останов алгоритма.

В результате на l-ых сегментах конвейера $(l = \overline{1, L})$ сформированы последовательности π_2^l одного из видов: 1) $\pi_2^l = \varnothing$, если ни один из типов данных $i \in N'_{пр}$ в π_2^l добавлен не был; 2) $\pi_2^l = \{i_{q+1}^l, i_{q+2}^l, ..., i_{q+n_l}^l\}$, где n_l – количество данных, которые были добавлены в последовательность π_2^l. После того как начальное решение по порядку данных $i \in N'_{пр}$ в последовательностях π_2^l $(l = \overline{1, L})$ сформировано, осуществляется определение на его основе эффективных порядков обработки данных. При этом последовательности π_2^l содержат данные в порядке возрастания значений идентификатора i; после формирования начального решения для последовательностей π_2^l количество данных в π^l $(l = \overline{1, L})$ одинаково. Для определения эффективных порядков обработки данных в последовательностях π_2^l на основе сформированного начального решения выполнена повторная инициализация множества $N'_{пр}$ следующим образом: $N'_{пр} = \bigcup_{l=1}^{L} \pi_2^l$, тогда множество $N'_{пр}$ – это типы данных, порядок которых в последовательностях π_2^l будет изменен. Сформированное множество $N'_{пр}$ упорядочивается по возрастанию идентификаторов типов данных $i \in N'_{пр}$.

Для определения эффективных порядков обработки данных в сформированных последовательностях π_2^l использован подход, предложенный в [2]–[5]. В соответствии с [2]–[5] рассматривается последовательность π_2^l, в которой на предварительном этапе размещено n_l данных и реализуется перемещение в ней данных из текущей позиции на новое местоположение (изменение местоположения данных в π_2^l), либо взаимная перестановка двух различных единичных данных в этой последовательности. Обозначим позиции данных рассматриваемого i'-го типа в последовательностях π_2^l как j'_l. Тогда в начальном решении π данные i'-го типа занимают в последовательностях π_2^l позиции j'_l, которые в процессе поиска эффективного решения будут изменяться. В результате для исходного начального решения π определяется окрестность $O_1(\pi)$, состоящая из решений, отличающихся от решения π тем, что в одной из последовательности π_2^l рассматриваемые данные i'-го типа (если они входят в последовательность) перемещаются на одну

позицию ближе к началу (выполняется изменение позиции j'_l данных i'-го типа). При одновременном изменении положения данных одного типа сразу в нескольких последовательностях π_2^l (по аналогии с [1]) может быть выполнен переход к окрестностям $O_k(\pi)$, при $k=\overline{1,k_{\max}}$ ($k_{\max}$ – максимальная метрика окрестности $O_k(\pi)$). По аналогии с [1] рассматриваемые данные i' перемещаются на одну позицию к началу в каждой последовательности π_2^l (при $i'\in\pi_2^l$) либо в совокупности последовательностей π_2^l, количество которых определяется индексом k ($k=\overline{1,k_{\max}}$). Затем для полученного нового локально эффективного решения формируются решения, входящие в его окрестности $O_k(\pi)$ ($k=\overline{1,k_{\max}}$), построенные в соответствии с рассмотренным подходом. Если в сформированных окрестностях $O_k(\pi)$ более эффективное решение не найдено, реализуется переход к следующему типу данных $i'\in N'_{np}$, позиции j'_l которых в π_2^l будет изменяться.

В соответствии с выполненными рассуждениями по построению новых решений (в окрестностях текущего решения π) сформулирован алгоритм определения эффективных порядков обработки данных, размещенных в π_2^l на первом этапе их формирования, который имеет следующий порядок шагов:

1) начальное решение по составам и порядку обработки данных в последовательностях π^l (последовательностях π_2^l, $l=\overline{1,L}$), полученное после размещения в π_2^l данных i-ых типов ($i\in N_{np}\cap(\bigcup_{l=1}^{L}\pi_1^l)$) определим как локально эффективное и обозначим как $\pi^*(s)$;

2) из множества N'_{np} выбирается тип данных i' в соответствии с условием: $i'=\min\{i\mid i\in N'_{np}\}$ (на текущем шаге алгоритма определяются эффективные местоположения данных i' в последовательностях π_2^l, $i'\in\pi_2^l$);

3) в соответствии с условием $i'\in\pi_2^l$ ($l=\overline{1,L}$) формируется множество Ls номеров сегментов, в последовательностях π_2^l которых находятся данные рассматриваемого типа i' (изменению подлежат позиции данных i'-го типа в тех последовательностях π_2^l, для которых $l\in Ls$ при выполнении условия $i'\in\pi_2^l$, возможные позиции рассматриваемых данных i' в π_2^l: $(q+1)$-я, $(q+2)$-я, …, $(q+n_l)$-я); определяются значения j'_l начальных позиций данных i' в последовательностях π_2^l ($l\in Ls$);

4) значение метрики k окрестности $O_k(\pi^*(s))$ решения $\pi^*(s)$, в которой будет выполняться поиск локально эффективных решений, задается равным 1;

5) значение g индекса промежуточного решения инициализируется 1 ($(s+g)$ – номер шага алгоритма, связанного с определением локального эффективного решения по местоположению данных i'-го типа в последовательностях π_2^l, формируемого на основе эффективного расписания $\pi^*(s)$ (соответствующее $\pi^*(s)$ решение $(P^l(s);\ (t_{ji}^{0l}(s)))^*$);

6) для каждого сегмента l (при условии $l\in Ls$) выполняется проверка условия $(q+1)<j'_l$ (данные i' занимают в $\pi^l(s)$ одну из следующих за $(q+1)$-ой позиций j'_l и их положение в π_2^l может быть изменено);

7) для l-ых сегментов конвейера, в последовательностях π_2^l которых выполняется условие $(q+1)<j'_l$, реализуется изменение положения данных i'-го типа в π_2^l ($l\in Ls$) на одну позицию ближе к началу последовательностей в зависимости от текущего значения окрестности $O_k(\pi(s))$; т. к. порядок данных в последовательностях π^l представляется матрицами P^l ($l=\overline{1,L}$), тогда изменение j'_l-ой позиции рассматриваемых данных в последовательностях π_2^l ($l\in Ls$) выполняется следующим образом:

$$p^l_{i',j'_l-1}(s+g)=1,\ \ p^l_{i',j'_l}(s+g)=0,$$

$$p^l_{k',j'_l-1}(s+g)=0,\ \ p^l_{k',j'_l}(s+g)=1,$$

где k' – индексы строк в матрицах P^l ($l\in Ls$), в которых элементы p^l_{k',j'_l-1} в (j'_l-1)-ом столбце равны 1 (на s-ом шаге алгоритма – это предыдущая позиция для рассматриваемых данных, которую они занимают на $(s+g)$-ом шаге); формируются матрицы $P^l(s+g)$ ($l\in Ls$), количество сформированных на $(s+g)$-ом шаге алгоритма

матриц $(P^l(s+g))$ определяется размером окрестности $O_k(\pi^*(s))$;

8) с использованием матриц P^l $(l=\overline{1,L})$ вычисляются матрицы (t_{ji}^{0l}) $(l=\overline{1,L})$ на $(s+g)$-ом шаге алгоритма, а также значения $f(s+g)$ (в итоге формируются решения $((P^l(s+q)),(t_{ji}^{0l}(s+q))\,|\,l=\overline{1,L})$);

9) для $((P^l(s)),(t_{ji}^{0l}(s))\,|\,l=\overline{1,L})^*$ и каждого решения $((P^l(s+g)),(t_{ji}^{0l}(s+g))\,|\,l=\overline{1,L})$ вычисляются значения левых ${}^-\nabla_g f(s)$ либо правых ${}^+\nabla_g f(s)$ дискретных градиентов [1]:

${}^-\nabla_g f(s)=[f(s+g)-f(s)]\le 0,$

${}^+\nabla_g f(s)=[f(s+g)-f(s)]>0;$ в результате индексы g групп последовательностей π^l, для которых ${}^-\nabla_g f(s)\le 0$ добавляются в множество N_s^p: $N_s^p=N_s^p\cup\{g\}$, формируется множество N_s^p направлений g, для которых ${}^-\nabla_g f(s)\le 0$; при $N_s^p=\varnothing$ реализуется переход к шагу 14;

10) в множестве N_s^p определяется направление g', для которого $\left|{}^-\nabla_g f(s)\right|$ наибольший: $g'\to\max\limits_g(|\,{}^-\nabla_g f(s)\,|)$; решение $((P^l(s+g')),(t_{ji}^{0l}(s+g'))\,|\,l=\overline{1,L})$ рассматривается как локально эффективное, на основе которого формируются последующие решения;

11) полученное решение $((P^l(s+g')),(t_{ji}^{0l}(s+g'))\,|\,l=\overline{1,L})$ фиксируется как локально эффективное: $((P^l),(t_{ji}^{0l})\,|\,l=\overline{1,L})^*=((P^l(s+g')),(t_{ji}^{0l}(s+g'))\,|\,l=\overline{1,L})$, фиксируется значение номера шага алгоритма $s=s+g$; индекс j'_l текущего столбца, идентифицирующий местоположение в π_2^l рассматриваемых данных i'-го типа для последовательностей, измененных на текущем шаге, модифицируется: $j'_l=j'_l-1$;

12) выполняется проверка условия $(q+1)<j'_l$ для каждой последовательности π_2^l $(l\in Ls)$;

13) при не выполнении условия $(q+1)<j'_l$ для последовательностей π_2^l $(l\in Ls)$, в которых было осуществлено изменение порядка данных на $(s+g)$-ом шаге алгоритма, индексы этих последовательностей l исключаются из множества Ls: $Ls=Ls\setminus\{l\}$, где номера l соответствуют последовательностям, входящим в группу, идентифицируемую индексом g', и для них выполняется условие $j'_l=(q+1)$, при выполнении условия $Ls\ne\varnothing$ реализуется переход к шагу 7; при $Ls=\varnothing$ отсутствуют последовательности π^l (последовательности π_2^l, $l\in Ls$), в которых может быть изменен порядок обработки данных, выполняется переход к шагу 16;

14) если $N_s^p=\varnothing$, то в окрестности O_k локально эффективного решения $((P^l(s)),(t_{ji}^{0l}(s))\,|\,l=\overline{1,L})^*$ (расписания $\pi^*(s)$) новое решение, уменьшающее значение целевой функции, не найдено, тогда значение метрики окрестности $O_k(\pi^*(s))$ модифицируется: $k=k+1$, если $k\le k_{\max}$, то выполняется переход к шагу 6;

15) при выполнении условия $k>k_{\max}$ в окрестности O_k текущего локально эффективного решения $((P^l(s)),(t_{ji}^{0l}(s))\,|\,l=\overline{1,L})^*$ лучшее решение не найдено; тогда решение $((P^l(s)),(t_{ji}^{0l}(s))\,|\,l=\overline{1,L})^*$ интерпретируется как эффективное решение по определению позиций данных i'-го типа в π^l $(l=\overline{1,L})$ (эффективное решение по определению позиций данных i'-го типа в π^l $(l=\overline{1,L})$); при реализации условия $k>k_{\max}$ выполняется переход к шагу 16;

16) реализуется модификация множества N'_{np} следующим образом: $N'_{np}=N'_{np}\setminus\{i'\}$, если для полученного в результате модификации множества N'_{np} выполняется условие $N'_{np}=\varnothing$, тогда все данные i-ых типов $(i\in N'_{np})$ размещены в последовательностях $(l=\overline{1,L})$ эффективным образом, тогда должен быть выполнен переход на шаг 17; если $N'_{np}\ne\varnothing$, тогда реализуется переход на шаг 2;

17) останов алгоритма.

В результате реализации сформулированного алгоритма в последовательностях $(l=\overline{1,L})$ (в последовательностях π_2^l) эффективным образом определены позиции данных i-ых типов, для которых перед построением динамического расписания выполнялось: $i\in N_{np}\cap(\bigcup\limits_{l=1}^{L}\pi_1^l)$. На втором этапе построения динамического расписания реализуется определение эффективного вида последовательностей π_2^l $(l\in Ls)$ при размещении в них дан-

ных i-ых типов, для которых перед началом построения динамического расписания выполнялись условия: $i \in N_{пр}\ \&\ (i \notin \bigcup_{l=1}^{L} \pi_1^l)$. Те же i-ые типы данных входят в множество $N_{пр}$, полученное после реализации приведенного выше алгоритма размещения в последовательностях π_2^l данных, для типов i которых выполняется условие $i \in N_{пр} \cap (\bigcup_{l=1}^{L} \pi_1^l)$. Так как данные, типы которых входят в полученное после реализации первого этапа алгоритма построения динамического расписания множество $N_{пр}$, должны быть добавлены в каждую из последовательностей π_2^l ($l = \overline{1, L}$), тогда для определения их местоположения (позиции j) в этих последовательностях применен «жадный» алгоритм, предложенный в [1].

При исследовании особенностей построения динамических расписаний решалась задача с пятью сегментами конвейера $(L = 5)$ и пятью типами обрабатываемых данных $(n = 5)$. При увеличении количества типов данных тенденция изменения эффективности формирования динамического расписания сохраняется. В качестве отказавшего задан второй сегмент конвейера. Длительности обработки данных на первом, четвертом и пятом сегментах заданы одинаковыми. Для определения степени несогласованности длительностей обработки данных на отказавшем и предшествующем ему сегментах в рассмотрение введено отношение: $\max_i(t_i^2) / \min_i(t_i^l)$, где t_i^2 – длительности обработки данных i-ых типов на $(l = 2)$-ом сегменте, t_i^l – длительности обработки данных i-ых типов на $(l = 1)$-ом сегменте конвейера. При проведении экспериментов значения этого отношения изменялись от 1,5 до 4 с шагом 0,5. Для определения степени несогласованности длительностей обработки данных на отказавшем и следующем за ним сегментах в рассмотрение введено отношение вида: $(\max_i(t_i^3) / \max_i(t_i^2))$, где t_i^3 – длительности обработки данных i-ых типов на $(l = 3)$-ом сегменте конвейера. При проведении экспериментов значения этого отношения изменялись от 2 до 5 с шагом 1. При этом $\min_i(t_i^3) = \min_i(t_i^2) = \min_i(t_i^l)$. Длительность восстановления отказавшего сегмента определялась как $t_{i_{вос}}^2 = k * \min_i(t_i^l)$, где k принимает значения 1, 3, 5, 7, 9, 11. Результаты формирования расписаний представлены в виде диаграмм Ганта на рис. 2, 3. Результаты исследований по определению эффективности расписаний в абсолютных временных единицах (уменьшение суммарных простоев сегментов конвейера при построении динамического расписания по сравнению с простоями сегментов конвейера для статического расписания при учете отказа сегмента) представлены на рис. 4–13. На основе значений, характеризующих эффективность метода построении динамического расписания (в единицах времени), определена относительная эффективность метода. Относительная эффективность метода представляет собой отношение абсолютной эффективности метода в временных единицах (улучшение значений критерия при построении динамического расписания по сравнению с статическим расписанием, учитывающим отказ сегмента) к общей (суммарной) величине простоев сегментов конвейера при обработке данных, вычисляемой для статического расписания в соответствии с выражением (4). Для каждой из зависимостей (рис. 4–13), реализовано построение зависимостей относительной эффективности метода от входных параметров в решаемой задаче, которые представлены на рис. 14–23. Анализ зависимостей относительной эффективности метода построения динамических расписаний при учете отказа позволяет сделать вывод о том, что этот метод реализует улучшение статического расписания, при этом улучшение значений критерия эффективности вида (4) осуществляется в диапазоне от 5 до 20 %. В тоже время анализ относительной эффективности позволяет сделать вывод о том, что при неизменных значениях входных параметров задачи, но при увеличении длительности восстановления отказавшего сегмента эффективность метода снижается.

ЗАКЛЮЧЕНИЕ

Результатами выполненных научных исследований являются:

1) сформулированные условия формирования последовательностей данных в динамическом расписании, для которых может быть изменен порядок обработки с учетом отказа сегмента конвейера и для которых порядок обработки изменен быть не может;

2) разработанный способ формирования последовательностей данных на соответствующих сегментах конвейера, которые будут оптимизированы предложенным градиентным методом составления расписаний;

3) разработанные способы определения эффективных порядков обработки данных в последовательностях на соответствующих сегментах конвейера при учете отказа сегмента;

4) определенные значения входных параметров задачи, при которых рассматриваемый метод построения динамических расписаний позволяет получить максимально эффективные решения.

СПИСОК ЛИТЕРАТУРЫ

1. *Кротов К. В.* Жадный алгоритм построения расписаний обработки данных в конвейерных системах / К. В. Кротов // Вестник Воронеж. гос. ун-та. Сер. Системный анализ и информационные технологии. – 2015. – № 1. – С. 44–59.

2. *Кочетов Ю. А.* Локальный поиск с чередующимися окрестностями / Ю. А. Кочетов, Н. Младенович, П. Хансен // Дискретный анализ и исследование операций. – Т. 10, № 1. – 2003. – С. 11–43.

3. *Кононова П. А.* Нижние и верхние оценки длины оптимального расписания презентаций медиа-объектов / П. А. Кононова // Дискретный анализ и исследование операций. – Том 19, № 1. – 2012. – С. 54–73.

4. *Кононова П. А.* Локальный поиск с чередующимися окрестностями для задачи Джонсона с пассивным буфером / П. А. Кононова // Дискретный анализ и исследование операций. – Т. 19, №5. – 2012.– с. 63–82.

5. *Кононова П. А.* Локальный поиск с чередующимися окрестностями для задачи Джонсона с пассивным буфером / П. А. Кононова // Диссертация кандидата физ.-мат. наук. – Новосибирск, Ин-т математики им. Соболева С. Л., 2012. – 106 с.

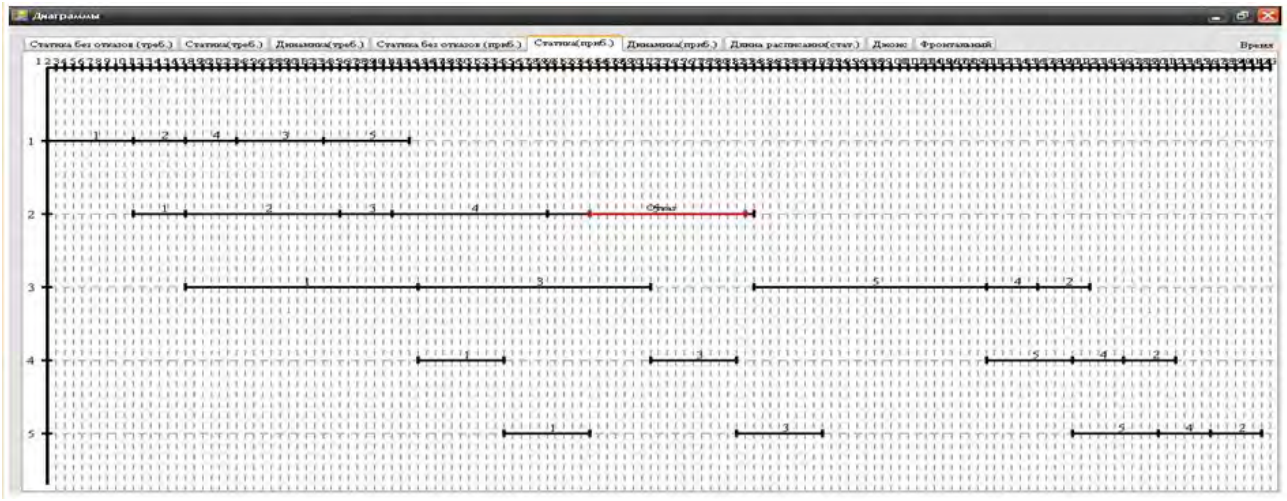

Рис. 2. Вид последовательностей обработки данных в статическом расписании при учете отказа сегмента (отказ $(l = 2)$ *-го сегмента)*

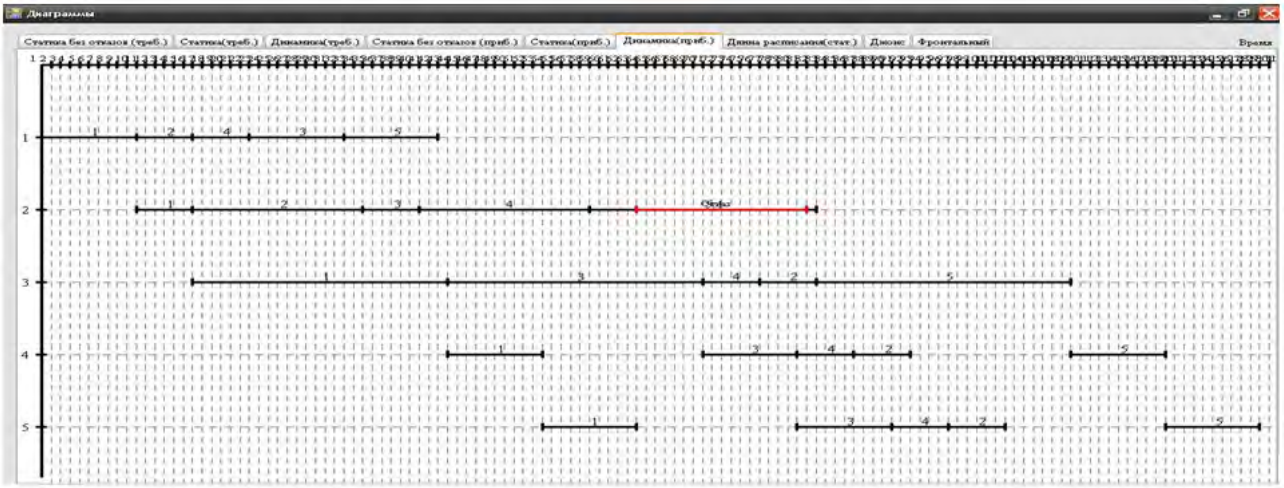

Рис. 3. Вид последовательностей обработки данных в динамическом расписании при учете отказа сегмента (отказ $(l = 2)$ -го сегмента)

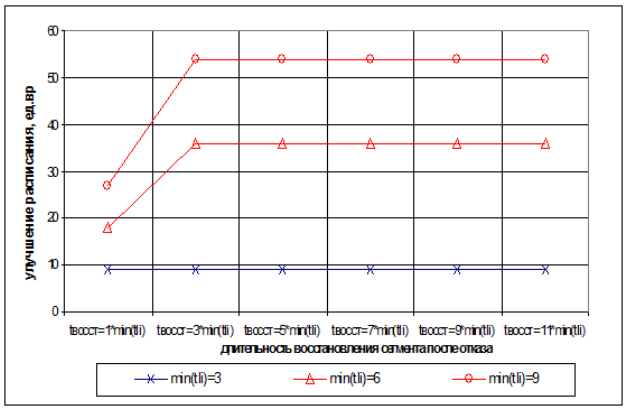

Рис. 4. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^l) = 2)$ и $(\max(t_i^3)/\max(t_i^2) = 1.5)$

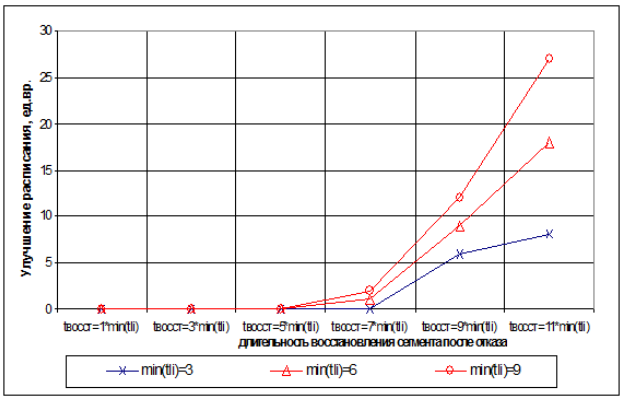

Рис. 6. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^l) = 4)$ и $(\max(t_i^3)/\max(t_i^2) = 1.5)$

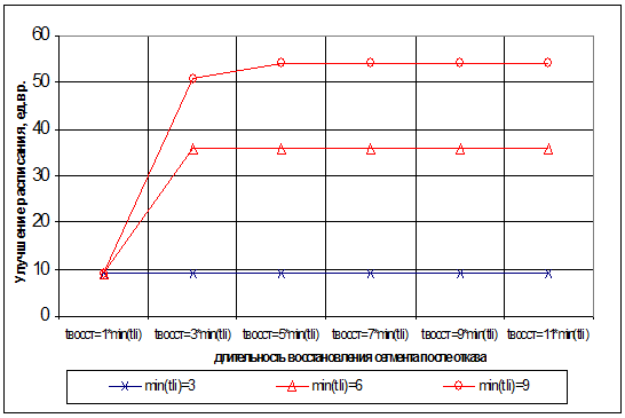

Рис. 5. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^l) = 3)$ и $(\max(t_i^3)/\max(t_i^2) = 1.5)$

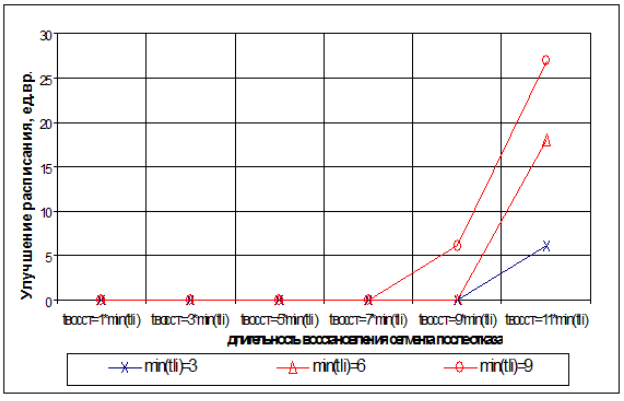

Рис. 7. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^l) = 5)$ и $(\max(t_i^3)/\max(t_i^2) = 1.5)$

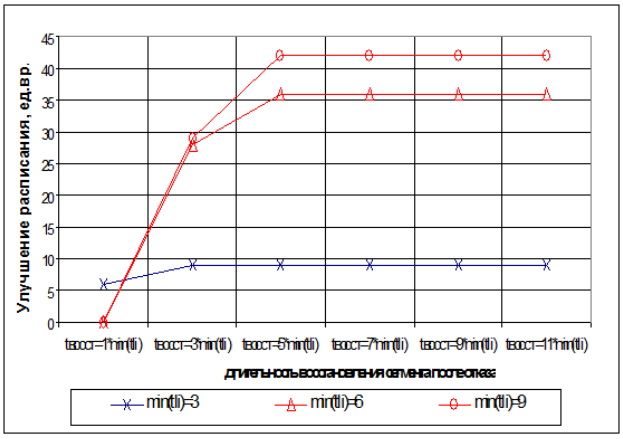

Рис. 8. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^1)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=2)$

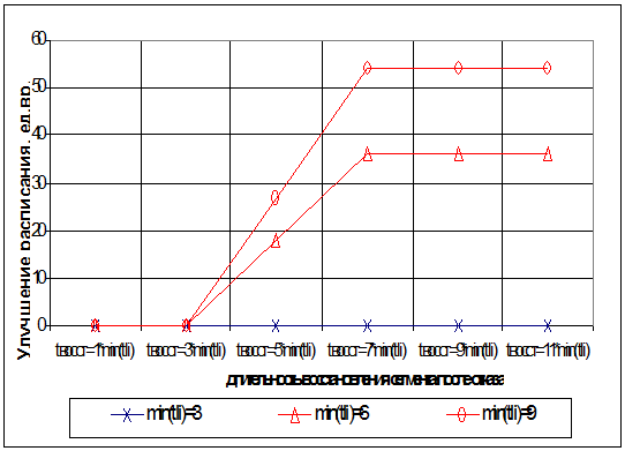

Рис. 9. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^1)=3)$ *и* $(\max(t_i^3)/\max(t_i^2)=2)$

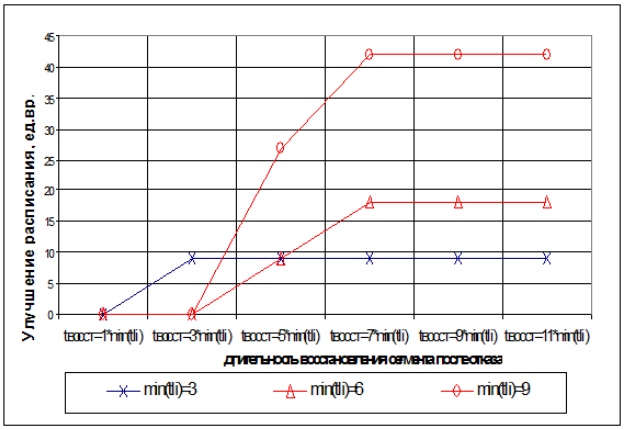

Рис. 10. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^1)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=2.5)$

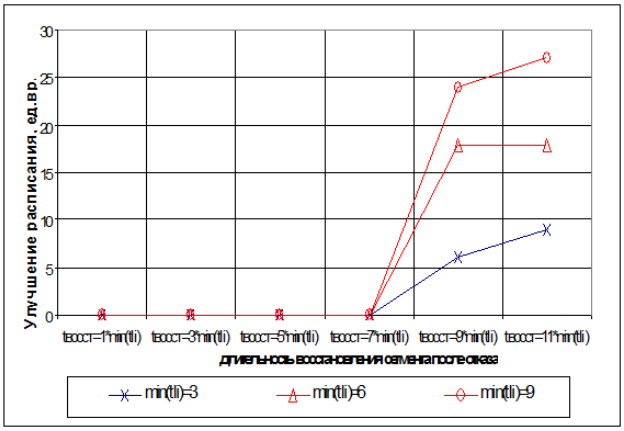

Рис. 11. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^1)=3)$ *и* $(\max(t_i^3)/\max(t_i^2)=2.5)$

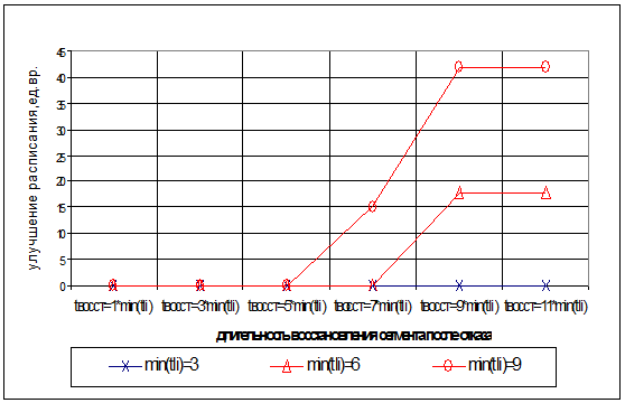

Рис. 12. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^1)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=3)$

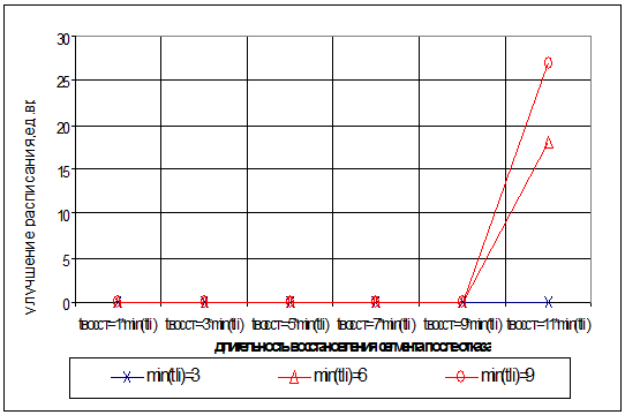

Рис. 13. Эффективность метода построения динамических расписаний при $(\max(t_i^2)/\min(t_i^1)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=3.5)$

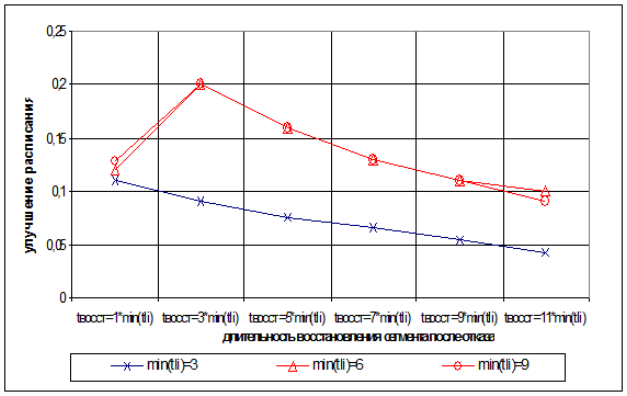

Рис. 14. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^l)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=1.5)$

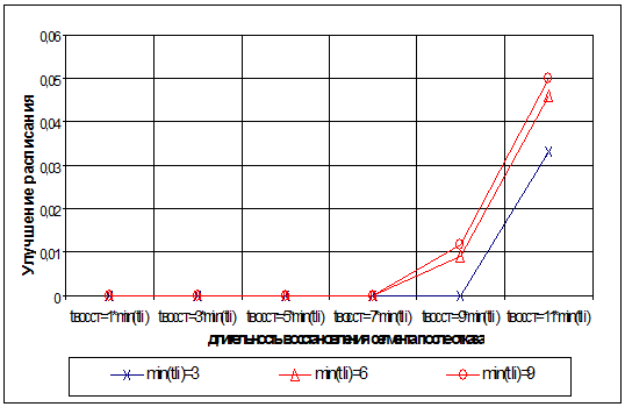

Рис. 17. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^l)=5)$ *и* $(\max(t_i^3)/\max(t_i^2)=1.5)$

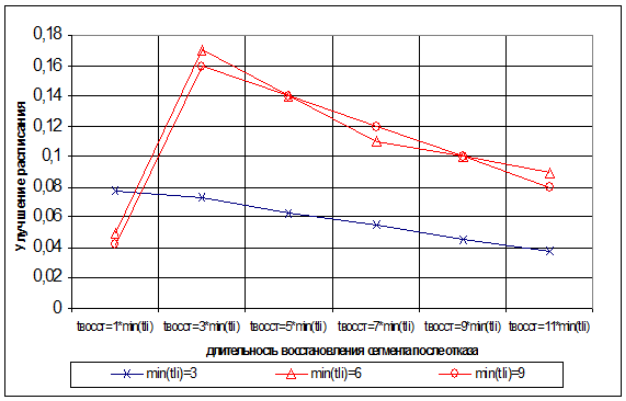

Рис. 15. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^l)=3)$ *и* $(\max(t_i^3)/\max(t_i^2)=1.5)$

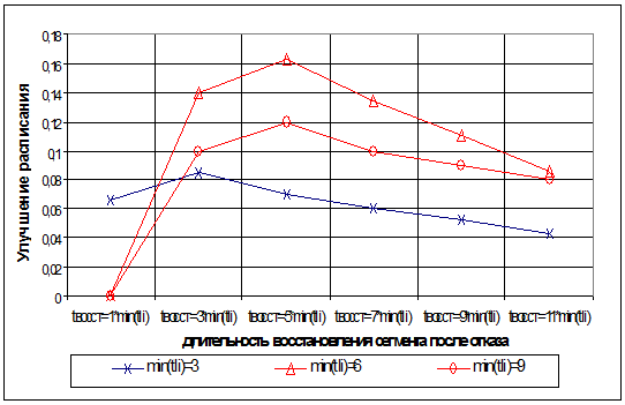

Рис. 18. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^l)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=2)$

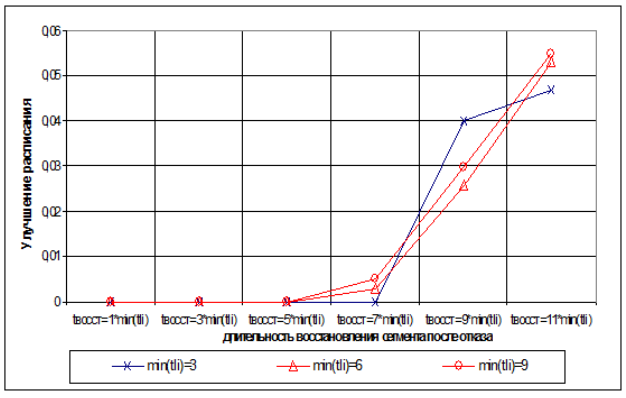

Рис. 16. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^l)=4)$ *и* $(\max(t_i^3)/\max(t_i^2)=1.5)$

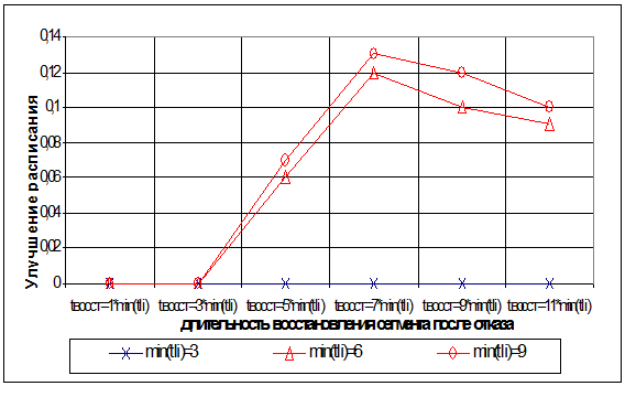

Рис. 19. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^l)=3)$ *и* $(\max(t_i^3)/\max(t_i^2)=2)$

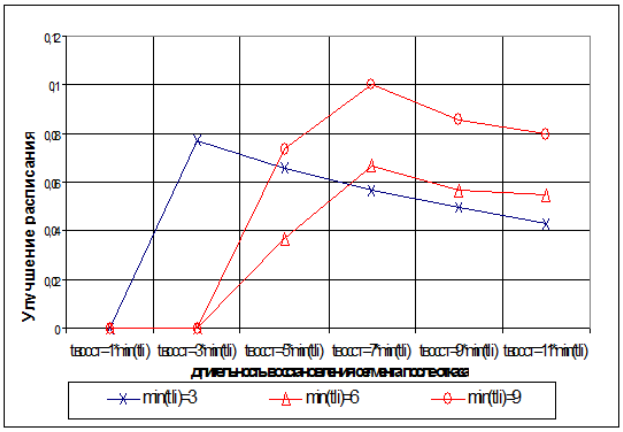

Рис. 20. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^1)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=2.5)$

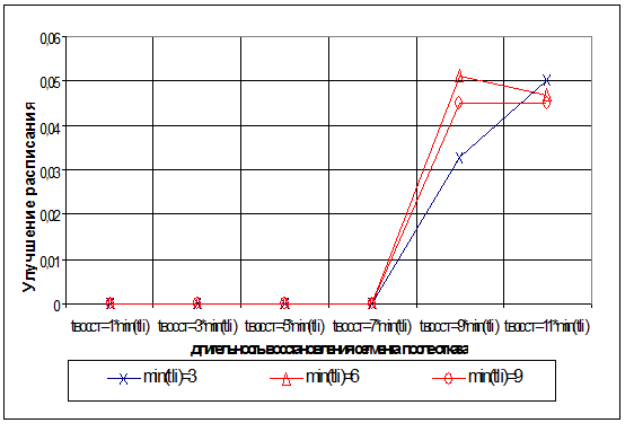

Рис. 21. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^1)=3)$ *и* $(\max(t_i^3)/\max(t_i^2)=2.5)$

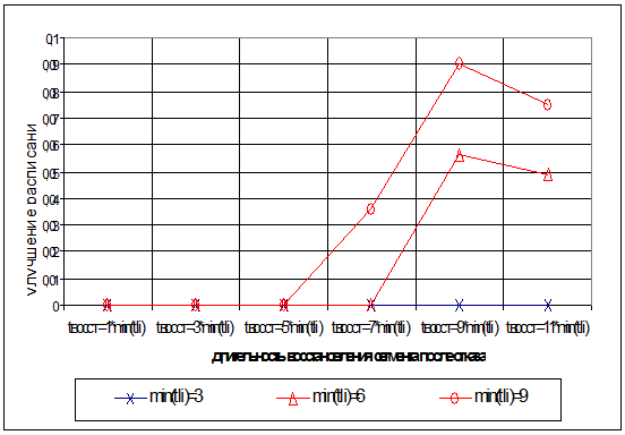

Рис. 22. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^1)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=3)$

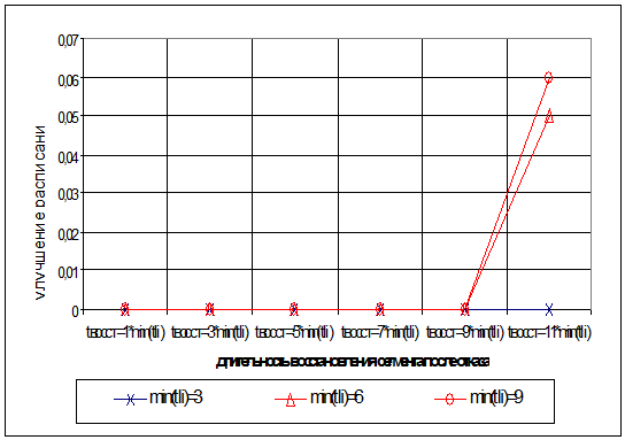

Рис. 23. Относительная эффективность динамического расписания при $(\max(t_i^2)/\min(t_i^1)=2)$ *и* $(\max(t_i^3)/\max(t_i^2)=3.5)$

Кротов К. В. – доцент, канд. техн. наук, Севастопольский Государственный университет, Севастополь, Россия
krotov_k1@mail.ru

Кротова Т. Ю. – старший преподаватель, Севастопольский Государственный университет, Севастополь, Россия

Krotov Kirill – Candidat of technical sciences, Associate Professor, Department of Information Systems Institute of Information Technology and Management in technical systems, Sevastopol State University.

Krotova Tatyana – Senior Lecturer, Department of Higher Mathematics Institute of Information Technology and Management in technical systems, Sevastopol State University.